



Manual do usuário

AWS CloudHSM



AWS CloudHSM: Manual do usuário

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

As marcas comerciais e imagens de marcas 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 desprestige 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 AWS CloudHSM é	1
Casos de uso	2
Como funciona	4
Clusters	5
Usuários em AWS CloudHSM	6
Entrada de chaves AWS CloudHSM	6
Cliente SDKs	8
Backups	8
Regiões suportadas para AWS CloudHSM	10
Preços para AWS CloudHSM	10
Introdução	11
Criar administradores do IAM	11
Criar um usuário do IAM e um grupo de administradores do IAM	12
Crie uma VPC	14
Criar um cluster	15
Rever grupo de segurança do cluster	19
Iniciar um cliente do EC2	20
Configurar o grupo de segurança da instância EC2	23
Etapa 1. Modificar o grupo de segurança padrão	23
Etapa 2. Conecte a instância do Amazon EC2 ao cluster AWS CloudHSM	24
Criar um HSM	25
Verificar a identidade do HSM (opcional)	27
Etapa 1. Obter certificados do HSM	29
Etapa 2. Obter certificados raiz	32
Etapa 3. Verificar cadeias de certificados	33
Etapa 4: Extrair e comparar chaves públicas	34
Inicializar o cluster	35
Visão geral do	35
Etapa 1. Obter a Solicitação de assinatura de certificado (CSR) do cluster	36
Etapa 2. Crie uma chave privada para sua CA raiz	38
Etapa 3. Assinar a CSR	39
Etapa 4: Inicializar o cluster	49
Instalar a CLI do CloudHSM	51
Ativar o cluster	55

Configurar mTLS (recomendado)	59
Etapa 1. Criar e registrar uma âncora de confiança no HSM	59
Etapa 2. Habilite o mTLS para AWS CloudHSM	63
Etapa 3. Defina a aplicação do mTLS para AWS CloudHSM	68
Crie e use chaves em AWS CloudHSM	70
Práticas recomendadas	72
Gerenciamento de clusters	72
Escale seu cluster para lidar com picos de tráfego	72
Arquitete seu cluster para alta disponibilidade	72
Tenha pelo menos três HSMs para garantir a durabilidade das chaves recém-geradas	73
Acesso seguro ao cluster	73
Reduza os custos escalando de acordo com suas necessidades	73
Gerenciamento de usuários	74
Proteja as credenciais de seus usuários do HSM	74
Tenha pelo menos dois administradores para evitar o bloqueio	74
Habilite o quórum para todas as operações de gerenciamento de usuários	74
Crie vários usuários de criptografia, cada um com permissões limitadas	75
Gerenciamento de chaves	75
Escolha o tipo de chave certo	75
Gerencie os principais limites de armazenamento	75
Gerenciando e protegendo o encapsulamento de chaves	76
Integração de aplicativos	77
Faça o bootstrap do seu Client SDK	77
Autentique-se para realizar operações	77
Gerencie com eficácia as chaves em seu aplicativo	78
Use multienlaceamento	79
Lide com erros de controle de utilização	79
Integre novas tentativas em operações de cluster	79
Implementação de estratégias de recuperação de desastres	80
Escalone a implantação de seu aplicativo	80
Monitoramento	81
Monitoramento de logs do cliente	81
Monitoramento de logs de auditoria	81
Monitorar AWS CloudTrail	81
Monitore as CloudWatch métricas da Amazon	82
Clusters	83

Arquitetura do cluster	83
Sincronização do cluster	85
Alta disponibilidade e balanceamento de carga do cluster	86
Modos de cluster	87
Tipos de HSM	88
Conectar-se ao cluster	90
Coloque um certificado de emissão em cada instância do EC2	90
Especifique o local do certificado de emissão.	90
Bootstrap o Client SDK	93
Dimensionamento HSMs	96
Adicionar um HSM	97
Remover um HSM	99
Excluir um cluster	100
Criar clusters de backups	101
Criar clusters de backups (console)	101
Criar clusters de backups (AWS CLI)	102
Crie clusters a partir de backups (AWS CloudHSM API)	103
Migração de tipos de cluster do HSM	103
Migrar de hsm1.medium para hsm2m.medium	104
Usuários do HSM	112
Gerenciamento de usuários com a CLI do CloudHSM	112
Pré-requisitos	113
Tipos de usuário	115
Tabela de permissões	116
Criar administrador	118
Criar CUs	119
Listar todos os usuários	120
Alterar senhas	121
Excluir usuários	123
Gerenciar a MFA de usuários	124
Gerenciar autenticação de quórum (M de N)	139
Gerenciamento de usuários com o CMU	164
Pré-requisitos	165
Tipos de usuário	169
Tabela de permissões	171
Criar usuários	173

Listar todos os usuários	175
Alterar senhas	177
Excluir usuários	180
Gerenciar a 2FA para usuários	182
Como usar o CMU para gerenciar a autenticação de quórum	191
Chaves	212
Sincronização e durabilidade de chaves	212
Conceitos	213
Noções básicas sobre sincronização de chaves do	214
Alterar as configurações de durabilidade de chaves do cliente	215
Sincronização de chaves em clusters clonados do	221
Encapsulamento de chaves AES	221
Algoritmos compatíveis	222
Usando o encapsulamento de teclas AES AWS CloudHSM	224
Chaves confiáveis	225
Noções básicas sobre chaves confiáveis	226
Atributos de chave confiáveis	226
Como usar chaves confiáveis para agrupar chaves de dados	227
Como desagrupar uma chave de dados com uma chave confiável	230
Gerenciamento de chaves com a CLI do CloudHSM	231
Gerar chaves	231
Excluir chaves	238
Compartilhar e descompartilhar chaves	240
Filtrar por chaves	248
Marcar uma chave como confiável	255
Gerenciar autenticação de quórum (M de N)	257
Gerenciamento de chaves com o KMU	278
Gerar chaves	279
Importar chaves	280
Exportar chaves	283
Excluir chaves	285
Compartilhar e descompartilhar chaves	285
Marcar uma chave como confiável	286
Backups do cluster	287
Trabalhar com backups	287
Remover chaves expiradas ou usuários inativos	288

Considerar a recuperação de desastres	288
Excluir backups	288
Restaurar backups	290
Configurar a retenção de backup	291
Retenção de backup gerenciada	292
Copiar um backup entre regiões	295
Copiar backups para diferentes regiões (console)	296
Copiar backups para diferentes regiões (AWS CLI)	296
Copiar backups para diferentes regiões (AWS CloudHSM API)	296
Trabalhar com backups compartilhados	297
Pré-requisitos para compartilhar backups	297
Compartilhar um backup	298
Cancelar o compartilhamento de um backup	301
Identificar um backup compartilhado	302
Permissões para backups compartilhados	302
Faturamento e medição	303
Clusters clonados	304
Obter um endereço IP para um HSM	305
Tópicos relacionados	306
Marcar recursos	307
Adicionar ou atualizar tags	307
Listar tags	309
Remover marcações	309
Ferramentas da linha de comando	311
Configure a ferramenta	312
Ferramenta de configuração do Client SDK 5	313
Ferramenta de configuração do Client SDK 3	347
CloudHSM CLI	355
Plataformas compatíveis	356
Introdução	357
Modos de comando	365
Atributos de chaves	367
Configurações avançadas	374
Referência	380
AWS CloudHSM Utilidade de gerenciamento	638
Plataformas compatíveis	639

Introdução	640
Instalar o cliente (Linux)	644
Instalar o cliente (Windows)	648
Referência	649
Key Management Utility (KMU – utilitário de gerenciamento de chaves)	710
Introdução	710
Instalar o cliente (Linux)	715
Instalar o cliente (Windows)	717
Referência	718
Cliente SDKs	848
Verificar a versão	849
Comparar suporte para componentes	851
Biblioteca PKCS #11	852
CloudHSM Management Utility (CMU – utilitário de gerenciamento do CloudHSM)	852
Key Management Utility (KMU – utilitário de gerenciamento de chaves)	852
Provedor JCE	852
Mecanismo dinâmico do OpenSSL	853
Provedor de armazenamento de chaves (KSP)	853
Migrar para o SDK mais recente	854
Migrar a biblioteca PKCS #11	854
Migrar OpenSSL Dynamic Engine	859
Migrar o Provedor de armazenamento de chaves (KSP)	862
Migração do provedor JCE	870
Client SDK 5	882
Benefícios do SDK mais recente	883
Plataformas compatíveis	884
Biblioteca PKCS #11	887
Mecanismo dinâmico do OpenSSL	938
Provedor OpenSSL	946
Provedor de armazenamento de chaves (KSP)	953
Provedor JCE	985
Versão anterior	1025
Fazer upgrade do Client SDK 3	1026
Plataformas compatíveis	1035
Biblioteca PKCS #11	1037
Mecanismo dinâmico do OpenSSL	1080

Provedor JCE	1084
Provedores de KSP e CNG	1115
Integrar aplicativos de terceiros	1128
Descarregamento de SSL/TLS	1128
Como funciona	1129
Descarregue no Linux com o OpenSSL Engine	1131
Descarregue no Linux com o OpenSSL Provider	1200
Descarregamento no Linux com JSSE	1230
Descarregar no Windows	1241
Adicione um balanceador de carga (opcional)	1257
CA do Windows Server	1264
Client SDK 5 com Windows Server CA	1264
Client SDK 3 com Windows Server CA	1269
Criptografia do Oracle Database	1274
Configurar pré-requisitos	1276
Etapa 3: gerar a chave de criptografia mestra do Oracle TDE	1277
Microsoft SignTool	1279
SDK 5 do cliente com Microsoft SignTool	1279
SDK de cliente 3 com Microsoft SignTool	1283
Java Keytool e Jarsigner	1288
Client SDK 5 com Java Keytool e Jarsigner	1288
Client SDK 3 com Java Keytool e Jarsigner	1300
Ferramenta de geração e edição de manifestos da Microsoft	1318
Etapa 1: configurar os pré-requisitos	1318
Etapa 2: criar um certificado de assinatura	1319
Etapa 3: assinar um arquivo	1321
Outras integrações de fornecedores terceiros	1322
Monitoramento	1324
Logs do Client SDK	1324
Registro em log do Client SDK 5	1325
Registro em log do Client SDK 3	1326
AWS CloudTrail	1328
AWS CloudHSM informações em CloudTrail	1329
Entendendo as entradas do arquivo de AWS CloudHSM log	1330
Logs de auditoria	1331
Como funciona o registro em log	1332

Visualizar logs	1333
Interpretar logs	1335
Referência de log	1351
CloudWatch métricas	1354
desempenho	1356
Dados de desempenho	1356
Controle de utilização do HSM	1357
Segurança	1358
Controlar o acesso à API com políticas do IAM	1359
Atualize as políticas do IAM para IPv6	1359
Proteção de dados	1362
Criptografia em repouso	1364
Criptografia em trânsito	1364
End-to-end criptografia	1364
Backups do cluster	1365
Gerenciamento de identidade e acesso	1366
Conceder permissões usando políticas do IAM;	1367
Ações de API para AWS CloudHSM	1368
Chaves de condição para AWS CloudHSM	1369
Políticas predefinidas gerenciadas pela AWS para AWS CloudHSM	1369
Políticas gerenciadas pelo cliente para AWS CloudHSM	1369
Perfis vinculados ao serviço	1373
Validação de conformidade	1375
PINO PCI FAQs	1377
Defasagens	1378
Resiliência	1380
Segurança da infraestrutura	1380
Isolamento de rede	1381
Autorização dos usuários	1381
Endpoints da VPC (AWS PrivateLink)	1381
Considerações sobre AWS CloudHSM VPC endpoints	1382
Criação de uma interface VPC endpoint para AWS CloudHSM	1382
Criação de uma política de VPC endpoint para AWS CloudHSM	1382
Gerenciamento de atualizações	1383
Solução de problemas	1384
AWS CloudHSM problemas conhecidos	1384

Problemas conhecidos para todas as instâncias do HSM	1385
Problemas conhecidos de hsm1m.medium	1391
Problemas conhecidos do hsm2m.medium	1392
Problemas conhecidos da biblioteca do PKCS#11	1396
Problemas conhecidos do SDK do JCE	1402
Problemas conhecidos para o OpenSSL Dynamic Engine	1407
Problemas conhecidos do Key Storage Provider (KSP)	1410
Problemas conhecidos do OpenSSL Provider	1412
Problemas conhecidos para instâncias do Amazon EC2 que executam o Amazon Linux 2	1412
Problemas conhecidos para a integração de aplicativos de terceiros	1413
Problemas conhecidos da modificação de cluster	1413
Problemas conhecidos de falha de operação usando a versão 5.12.0 do AWS CloudHSM cliente no hsm2.medium	1414
Falhas de sincronização de chaves do Client SDK 3	1415
Verificar o desempenho com o Client SDK 3	1416
Teste das recomendações	1418
Opções configuráveis para a ferramenta pkpspeed	1418
Testes que podem ser executados com a ferramenta pkpspeed	1418
Exemplos	1419
O usuário do Client SDK 5 contém valores inconsistentes	1423
Falhas de replicação de usuário do Client SDK 5	1429
Problema: o usuário selecionado não está sincronizado em todo o cluster	1430
Problema: o usuário existe no cluster de destino com atributos diferentes	1431
Problema: falha na replicação do usuário de hsm2m.medium para hsm1.medium	1431
Falhas de replicação de chaves do Client SDK 5	1432
Problema: a chave selecionada não está sincronizada em todo o cluster	1432
Problema: existe uma chave com a mesma referência no cluster de destino com diferentes informações ou atributos	1434
AWS CloudHSM erro visto durante a verificação da disponibilidade da chave	1435
Extrair chaves usando o JCE	1436
getEncoded, getPrivateExponent, ou GETs retorna null	1436
getEncoded getPrivateExponent,, ou GETs retornam bytes de chave fora do HSM	1436
Controle de utilização do HSM	1436
Resolução	1437
Manter os usuários do HSM em sincronia	1438
Conexão perdida	1438

Registros AWS CloudHSM de auditoria ausentes CloudWatch	1442
Agrupamento de chaves AES não compatíveis	1442
Determine se seu código gera chaves agrupadas irrecuperáveis	1442
Ações que você deve realizar se seu código gerar chaves agrupadas irrecuperáveis	1444
Resolvendo falhas na criação de AWS CloudHSM clusters	1445
Adicionar a permissão ausente	1445
Criar a função vinculada ao serviço manualmente	1446
Usar um usuário não federado	1446
Recuperando registros de configuração AWS CloudHSM do cliente	1447
Ferramenta de suporte do Client SDK 5	1447
Ferramenta de suporte do Client SDK 3	1449
Cotas	1451
Downloads	1453
Versão mais recente	1453
Versão 5 do SDK do cliente: versão 5.17.0	1454
Versão anterior	1461
Versões descontinuadas	1496
Versões descontinuadas do Client SDK 5	1496
Versões descontinuadas do Client SDK 3	1511
End-of-life lançamentos	1521
Histórico do documento	1522
Atualizações recentes	1522
Atualizações anteriores	1528
.....	mdxxx

O que AWS CloudHSM é

AWS CloudHSM combina os benefícios da AWS nuvem com a segurança dos módulos de segurança de hardware (HSMs). Um módulo de segurança de hardware (HSM) é um dispositivo computacional que processa operações de criptografia e oferece armazenamento seguro para chaves criptográficas. Com AWS CloudHSM, você tem controle total sobre a alta disponibilidade HSMs que está na nuvem da AWS, tem acesso de baixa latência e uma raiz segura de confiança que automatiza o gerenciamento de HSM (incluindo backups, provisionamento, configuração e manutenção).

AWS CloudHSM oferece aos clientes uma variedade de benefícios:

Acesso a clusters FIPS e diferentes de FIPS

AWS CloudHSM oferece clusters em dois modos: FIPS e não FIPS. No modo FIPS, somente chaves e algoritmos validados pelo Federal Information Processing Standard (FIPS – Padrão federal de processamento de informações) podem ser usados. O modo não FIPS oferece todas as chaves e algoritmos suportados AWS CloudHSM, independentemente da aprovação do FIPS. Para obter mais informações, consulte [AWS CloudHSM modos de cluster](#).

HSMs são de uso geral, de inquilino único e validados para FIPS 140-2 de nível 3 ou FIPS 140-3 de nível 3 para clusters no modo FIPS

AWS CloudHSM usa propósitos gerais HSMs que fornecem mais flexibilidade quando comparados aos serviços totalmente gerenciados da AWS que têm algoritmos e comprimentos de chave predeterminados para seu aplicativo. Oferecemos soluções HSMs que estão em conformidade com os padrões, são de inquilino único e são validadas com FIPS 140-2 de nível 3 ou FIPS 140-3 de nível 3 para clusters no modo FIPS. Para clientes com casos de uso fora das restrições da validação do FIPS 140-2 ou FIPS 140-3 de nível 3, o AWS CloudHSM também oferece clusters no modo não FIPS. Consulte [AWS CloudHSM aglomerados](#) para obter mais informações.

A criptografia E2E não é visível para a AWS

Como seu plano de dados é criptografado end-to-end (E2E) e não é visível para a AWS, você controla seu próprio gerenciamento de usuários (fora das funções do IAM). A desvantagem desse controle é que você tem mais responsabilidade do que se usasse um serviço gerenciado da AWS.

Controle total de suas chaves, algoritmos e desenvolvimento de aplicativos

AWS CloudHSM oferece controle total dos algoritmos e das chaves que você usa. Você pode gerar, armazenar, importar, exportar e gerenciar chaves criptográficas (incluindo chaves de sessão, chaves de token, pares de chaves simétricas e assimétricas). Além disso, AWS CloudHSM SDKs oferece controle total sobre o desenvolvimento de aplicativos, a linguagem do aplicativo, o encadeamento e onde seus aplicativos existem fisicamente.

Migre suas workloads de criptografia para a nuvem

Os clientes que migram a infraestrutura de chave pública que usam os Padrões de Criptografia de Chave Pública #11 (PKCS #11), a Extensão Criptográfica Java (JCE), a API de Criptografia: Próxima Geração (CNG) ou o Provedor de Armazenamento de Chaves (KSP) podem migrar para o aplicativo com menos alterações. AWS CloudHSM

Para saber mais sobre o que você pode fazer com AWS CloudHSM, consulte os tópicos a seguir. Quando você estiver pronto para começar AWS CloudHSM, consulte [Introdução](#).

Note

Se você quiser um serviço gerenciado para criar e controlar suas chaves de criptografia, mas não quiser ou precisar operar suas próprias chaves HSMs, considere usá-lo [AWS Key Management Service](#).

Se você estiver procurando por um serviço elástico que HSMs gerencie pagamentos e chaves para aplicativos de processamento de pagamentos na nuvem, considere usar a [criptografia de pagamento da AWS](#).

Conteúdo

- [AWS CloudHSM casos de uso](#)
- [Como AWS CloudHSM funciona](#)
- [Preços para AWS CloudHSM](#)

AWS CloudHSM casos de uso

AWS CloudHSM pode ser usado para atingir uma variedade de objetivos. O conteúdo deste tópico fornece uma visão geral do que você pode fazer com AWS CloudHSM.

Conformidade regulatória

As empresas que precisam se alinhar aos padrões de segurança corporativos podem usar AWS CloudHSM para gerenciar chaves privadas que protegem dados altamente confidenciais. Os HSMs fornecidos pela AWS CloudHSM são certificados FIPS 140-2 nível 3 e estão em conformidade com o PCI DSS. Além disso, AWS CloudHSM é compatível com PCI PIN e PCI-3DS. Para obter mais informações, consulte [Validação de conformidade para AWS CloudHSM](#).

Criptografar e descriptografar dados

Use AWS CloudHSM para gerenciar chaves privadas que protegem dados altamente confidenciais, criptografia em trânsito e criptografia em repouso. Além disso, AWS CloudHSM oferece integração compatível com os padrões com vários criptográficos. SDKs

Assine e verifique documentos com chaves públicas e privadas

Na criptografia, o uso de uma chave privada para assinar um documento permite que os destinatários usem uma chave pública para verificar se você (e não outra pessoa) realmente enviou o documento. Use AWS CloudHSM para criar pares assimétricos de chaves públicas e privadas projetados especificamente para essa finalidade.

Autenticar mensagens usando HMACs e CMACs

Na criptografia, os códigos de autenticação de mensagens cifradas (CMACs) e os códigos de autenticação de mensagens baseados em hash (HMACs) são usados para autenticar e garantir a integridade das mensagens enviadas por redes inseguras. Com AWS CloudHSM, você pode criar e gerenciar com segurança chaves simétricas que suportam e. HMACs CMACs

Aproveite os benefícios de AWS CloudHSM e AWS Key Management Service

Os clientes podem combinar AWS CloudHSM e [AWS KMS](#) armazenar materiais essenciais em um ambiente de inquilino único e, ao mesmo tempo, obter os benefícios de gerenciamento de chaves, escalabilidade e integração na nuvem. AWS KMS Para obter detalhes sobre como fazer isso, consulte as [lojas de chaves do AWS CloudHSM](#) no Guia do desenvolvedor do AWS Key Management Service .

SSL/TLS Processamento de descarga para servidores web

Para enviar dados com segurança pela Internet, os servidores web usam pares de chaves pública-privada e certificado de chave SSL/TLS pública para estabelecer sessões HTTPS. Esse processo envolve muita computação para servidores web, mas você pode reduzir a carga computacional e, ao mesmo tempo, fornecer segurança extra transferindo parte disso para o seu cluster. AWS CloudHSM Para obter informações sobre como configurar o SSL/TLS offload with AWS CloudHSM, consulte. [Descarregamento de SSL/TLS](#)

Ativar criptografia de dados transparente (TDE)

Transparent Data Encryption [criptografia de dados transparente (TDE)] é usada para criptografar arquivos de banco de dados. Com o TDE, o servidor de banco de dados criptografa os dados antes de armazená-los no disco. Você pode obter maior segurança armazenando a chave de criptografia mestra do TDE HSMs em seu AWS CloudHSM. Para obter informações sobre como configurar o Oracle TDE com AWS CloudHSM, consulte [Criptografia do Oracle Database](#).

Gerenciar as chaves privadas para uma autoridade de certificado de emissão (CA)

Uma autoridade de certificação (CA) é uma entidade confiável que emite certificados digitais que vinculam uma chave pública a uma identidade (uma pessoa ou organização). Para operar uma CA, é necessário manter a confiança, protegendo as chaves privadas que assinam os certificados emitidos pela CA. Você pode armazenar essas chaves privadas em seu AWS CloudHSM cluster e depois usá-las HSMs para realizar operações de assinatura criptográfica.

Gere números randômicos

Gerar números aleatórios para criar chaves de criptografia é fundamental para a segurança online. AWS CloudHSM podem ser usados para gerar números aleatórios com segurança sob HSMs seu controle e só são visíveis para você.

Como AWS CloudHSM funciona

Este tópico fornece uma visão geral dos conceitos básicos e da arquitetura que você usa para criptografar dados com segurança e realizar operações criptográficas em. HSMs AWS CloudHSM opera em sua própria Amazon Virtual Private Cloud (VPC). Antes de poder usar AWS CloudHSM, primeiro você cria um cluster, adiciona HSMs a ele, cria usuários e chaves e, em seguida, usa o

Client SDKs para integrá-lo HSMs ao seu aplicativo. Feito isso, você usa os registros do SDK do cliente AWS CloudTrail, os registros de auditoria e CloudWatch a Amazon para [monitorar AWS CloudHSM](#).

Conheça AWS CloudHSM os conceitos básicos e saiba como eles funcionam juntos para ajudar a proteger seus dados.

Tópicos

- [AWS CloudHSM aglomerados](#)
- [Usuários em AWS CloudHSM](#)
- [Entrada de chaves AWS CloudHSM](#)
- [Cliente SDKs para AWS CloudHSM](#)
- [AWS CloudHSM backups de cluster](#)
- [Regiões suportadas para AWS CloudHSM](#)

AWS CloudHSM aglomerados

Fazer com que as pessoas HSMs trabalhem em conjunto de forma sincronizada, redundante e altamente disponível pode ser difícil, mas AWS CloudHSM faz o trabalho pesado para você, fornecendo módulos de segurança de hardware () em clusters. HSMs Um cluster é uma coleção de indivíduos HSMs que se AWS CloudHSM mantém sincronizados. Quando você executa uma tarefa ou operação em um HSM em um cluster, o outro HSMs nesse cluster é automaticamente atualizado.

AWS CloudHSM oferece clusters em dois modos: FIPS e não FIPS. No modo FIPS, somente chaves e algoritmos validados pelo Federal Information Processing Standard (FIPS – Padrão federal de processamento de informações) podem ser usados. O modo não FIPS oferece todas as chaves e algoritmos suportados AWS CloudHSM, independentemente da aprovação do FIPS. AWS CloudHSM também oferece dois tipos de HSMs: hsm1.medium e hsm2m.medium. Para obter detalhes sobre as diferenças entre cada tipo de HSM e modo de cluster, consulte [AWS CloudHSM modos de cluster](#). O tipo de HSM hsm1.medium está chegando ao fim do suporte, portanto não será possível criar novos clusters com esse tipo. Para saber mais, consulte [Notificações de suspensão de uso](#) para obter detalhes.

Para atender às suas metas de disponibilidade, durabilidade e escalabilidade, você define o número de HSMs em seu cluster em várias zonas de disponibilidade. Você pode criar um cluster que tenha de 1 a 28 HSMs (o [limite padrão](#) é de 6 HSMs por AWS conta por [AWS região](#)). Você pode colocá-los HSMs em diferentes [zonas de disponibilidade](#) em uma AWS região. Adicionar mais HSMs a um

cluster proporciona maior desempenho. Espalhar clusters pelas zonas de disponibilidade fornece redundância e alta disponibilidade.

Para obter mais informações sobre clusters, consulte [Clusters em AWS CloudHSM](#).

Para criar um cluster, consulte [Introdução](#).

Usuários em AWS CloudHSM

Ao contrário da maioria dos AWS serviços e recursos, você não usa usuários AWS Identity and Access Management (IAM) ou políticas do IAM para acessar recursos dentro do seu AWS CloudHSM cluster. Em vez disso, você usa usuários do HSM diretamente HSMs no seu AWS CloudHSM cluster.

Os usuários do HSM são diferentes dos usuários do IAM. Os usuários do IAM que têm as credenciais corretas podem criar interagindo com recursos HSMs por meio da API da AWS. Como a criptografia E2E não é visível para a AWS, você deve usar as credenciais de usuário do HSM para autenticar as operações no HSM, uma vez que as credenciais ocorrem diretamente no HSM. O HSM autentica cada usuário do HSM por meio das credenciais que você define e gerencia. Cada usuário do HSM tem um tipo que determina quais operações o usuário tem permissão para realizar no HSM. Cada HSM autentica cada usuário do HSM por meio das credenciais que você define usando a [CLI do CloudHSM](#).

Se você estiver usando a [série de versões anteriores do SDK](#), usará o [CloudHSM Management Utility \(CMU\)](#).

Entrada de chaves AWS CloudHSM

AWS CloudHSM permite que você gere, armazene e gerencie com segurança suas chaves de criptografia em um único local HSMs que estão em seu cluster. AWS CloudHSM As chaves podem ser simétricas ou assimétricas, podem ser chaves de sessão (chaves efêmeras) para sessões únicas, chaves de token (chaves persistentes) para uso a longo prazo e podem ser exportadas e importadas para. AWS CloudHSM As chaves também podem ser usadas para concluir tarefas e funções criptográficas comuns:

- Execute a assinatura de dados criptográficos e a verificação de assinaturas com algoritmos de criptografia simétrica e assimétrica.
- Trabalhe com funções hash para computar resumos de mensagens e códigos de autenticação de mensagens baseados em hash (). HMACs

- Encapsule e proteja outras chaves.
- Acessar dados aleatórios protegidos criptograficamente.

O máximo de chaves que um cluster pode ter depende do tipo HSMs que está no cluster. Por exemplo, o tipo hsm2m.medium armazena mais chaves do que o hsm1.medium. Para ver uma comparação, consulte [AWS CloudHSM cotas](#).

Além disso, AWS CloudHSM segue alguns princípios fundamentais para uso e gerenciamento de chaves:

Muitos tipos de chaves e algoritmos para escolher

Para permitir que você personalize suas próprias soluções, AWS CloudHSM fornece vários tipos de chaves e algoritmos para escolher, os algoritmos suportam uma variedade de tamanhos de chave. Para obter mais informações, consulte as páginas de atributos e mecanismos de cada um [Operações de descarga com o cliente AWS CloudHSM SDKs](#).

Como gerenciar chaves

AWS CloudHSM as chaves são gerenciadas por meio SDKs de ferramentas de linha de comando. Para obter informações sobre como usar essas ferramentas para gerenciar chaves, consulte [Entrada de chaves AWS CloudHSM](#) e [Melhores práticas para AWS CloudHSM](#).

Quem possui as chaves

Em AWS CloudHSM, o usuário criptográfico (UC) que cria a chave é o proprietário dela. O proprietário pode usar os key unshare comandos key share e para compartilhar e descompartilhar a chave com outras CUs pessoas. Para obter mais informações, consulte [Compartilhar e cancelar o compartilhamento de chaves usando a CLI do CloudHSM](#).

O acesso e o uso podem ser controlados com criptografia baseada em atributos

AWS CloudHSM permite que você use criptografia baseada em atributos, uma forma de criptografia que permite usar atributos-chave para controlar quem pode descriptografar dados com base em políticas.

Cliente SDKs para AWS CloudHSM

Ao usar AWS CloudHSM, você executa operações criptográficas com os [kits de desenvolvimento de software AWS CloudHSM do cliente \(SDKs\)](#). AWS CloudHSM O cliente SDKs inclui:

- Padrão de criptografia de chave pública N° 11 (PKCS #11)
- Provedor JCE
- Mecanismo dinâmico do OpenSSL
- Provedor de armazenamento de chaves (KSP) para Microsoft Windows

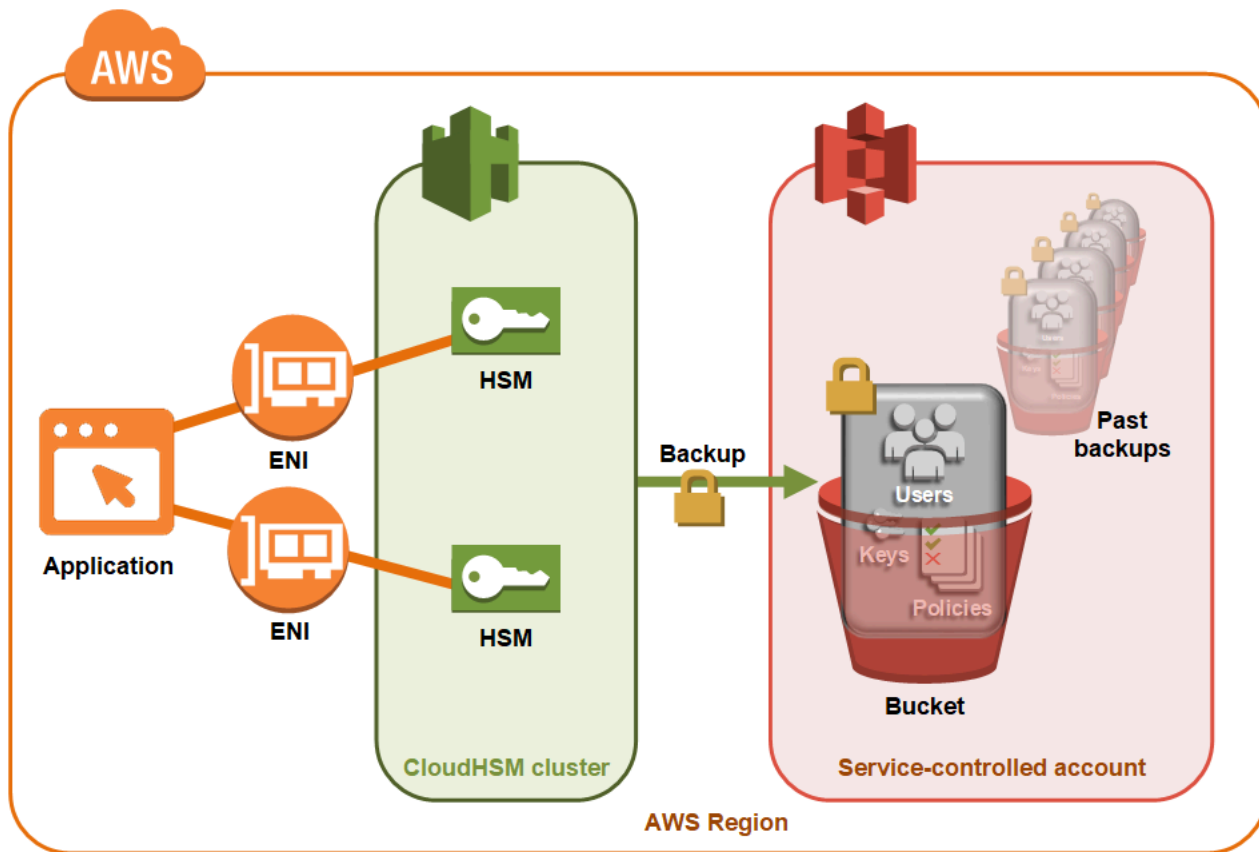
Você pode usar qualquer um ou todos esses SDKS em seu AWS CloudHSM cluster. Escreva o código do seu aplicativo para usá-lo SDKs para realizar operações criptográficas em seu HSMs. Para conferir quais plataformas e tipos de HSM são compatíveis com cada SDK, consulte [AWS CloudHSM Plataformas compatíveis com o Client SDK 5](#)

Ferramentas utilitárias e de linha de comando são necessárias não apenas para usar, SDKs mas também para configurar as credenciais, políticas e configurações do seu aplicativo. Para obter mais informações, consulte [AWS CloudHSM ferramentas de linha de comando](#).

Para obter mais informações sobre como instalar e usar o Client SDK ou a segurança da conexão do cliente, consulte [Cliente SDKs](#) e [End-to-end criptografia](#)

AWS CloudHSM backups de cluster

AWS CloudHSM faz backups periódicos dos usuários, chaves e políticas no cluster. Os backups são seguros, duráveis e atualizados em um cronograma previsível. A ilustração a seguir mostra o relacionamento entre seus backups e o cluster.



Para obter mais informações sobre Trabalhar com backups, consulte [Backups do cluster](#).

Segurança

Quando AWS CloudHSM faz um backup do HSM, o HSM criptografa todos os seus dados antes de enviá-los para. AWS CloudHSM Os dados nunca saem do HSM em formato de texto simples. Além disso, os backups não podem ser descriptografados AWS porque AWS não tem acesso à chave usada para descriptografar os backups. Para obter mais informações, consulte [Segurança dos backups do cluster](#).

Durabilidade

AWS CloudHSM armazena backups em um bucket do Amazon Simple Storage Service (Amazon S3) controlado pelo serviço na mesma região do seu cluster. Os backups têm um nível de durabilidade de 99,999999999%, o mesmo de qualquer objeto armazenado no Amazon S3.

Regiões suportadas para AWS CloudHSM

Para obter informações sobre as regiões suportadas AWS CloudHSM, consulte [AWS CloudHSM Regiões e endpoints](#) no Referência geral da AWS, ou na [tabela de regiões](#).

AWS CloudHSM pode não estar disponível em todas as zonas de disponibilidade em uma determinada região. No entanto, isso não deve afetar o desempenho, pois equilibra AWS CloudHSM automaticamente a carga em tudo HSMs em um cluster.

Como a maioria dos AWS recursos, HSMs agrupam e são recursos regionais. Não é possível reutilizar ou estender um cluster entre regiões. É obrigatório executar todas as etapas necessárias listadas em [Começando com AWS CloudHSM](#) para criar um cluster em uma nova região.

Para fins de recuperação de desastres, AWS CloudHSM permite que você copie backups do seu AWS CloudHSM cluster de uma região para outra. Para obter mais informações, consulte [AWS CloudHSM backups de cluster](#).

Preços para AWS CloudHSM

Com AWS CloudHSM, você paga por hora sem compromissos de longo prazo ou pagamentos antecipados. Para obter mais informações, consulte [AWS CloudHSM Preços](#) no AWS site.

Começando com AWS CloudHSM

Os tópicos a seguir ajudam você a criar, inicializar e ativar um cluster no AWS CloudHSM. Depois de concluir estes procedimentos, você estará pronto para gerenciar usuários e clusters, e usar as bibliotecas de software incluídas para executar operações de criptografia. Para obter a melhor experiência, siga os tópicos na ordem listada.

Conteúdo

- [Crie grupos administrativos do IAM para AWS CloudHSM](#)
- [Crie uma nuvem privada virtual \(VPC\) para AWS CloudHSM](#)
- [Crie um cluster em AWS CloudHSM](#)
- [Analise o grupo de segurança do seu cluster no AWS CloudHSM](#)
- [Inicie uma instância cliente do Amazon EC2 para interagir com AWS CloudHSM](#)
- [Configure os grupos de segurança da instância cliente Amazon EC2 para AWS CloudHSM](#)
- [Crie um HSM em AWS CloudHSM](#)
- [Verifique a identidade e a autenticidade do HSM do seu cluster em AWS CloudHSM \(opcional\)](#)
- [Inicialize o cluster em AWS CloudHSM](#)
- [Instalar e configurar a CLI do CloudHSM](#)
- [Ative o cluster em AWS CloudHSM](#)
- [Configurar o TLS mútuo entre o cliente e AWS CloudHSM \(recomendado\)](#)
- [Crie e use chaves em AWS CloudHSM](#)

Crie grupos administrativos do IAM para AWS CloudHSM

A primeira etapa para começar AWS CloudHSM é configurar as permissões do IAM.

Como [prática recomendada](#), não use o seu Usuário raiz da conta da AWS para interagir com AWS, inclusive AWS CloudHSM. Em vez disso, use AWS Identity and Access Management (IAM) para criar um usuário do IAM, uma função do IAM ou um usuário federado. Siga as etapas na seção [Criar um usuário do IAM e um grupo de administradores do IAM](#) para criar um grupo de administradores e anexar a AdministratorAccesspolítica a ele. Em seguida, crie outro usuário administrador e o adicione ao grupo. Adicione outros usuários ao grupo, conforme necessário. Cada usuário que você adiciona herda a AdministratorAccesspolítica do grupo.

Outra prática recomendada é criar um grupo de AWS CloudHSM administradores que tenha somente as permissões necessárias para execução AWS CloudHSM. Adicione usuários individuais a este grupo, conforme necessário. Cada usuário herdará as permissões limitadas anexadas ao grupo, em vez do acesso completo da AWS. A [Políticas gerenciadas pelo cliente para AWS CloudHSM](#) seção a seguir contém a política que você deve anexar ao seu grupo de AWS CloudHSM administradores.

AWS CloudHSM define uma [função vinculada ao serviço](#) para sua conta. Atualmente, a função vinculada ao serviço define permissões que permitem que sua conta registre eventos. AWS CloudHSM A função pode ser criada automaticamente AWS CloudHSM ou manualmente por você. Você não pode editar a função, mas pode excluí-la. Para obter mais informações, consulte [Funções vinculadas a serviços para AWS CloudHSM](#).

Criar um usuário do IAM e um grupo de administradores do IAM

Comece criando um usuário do IAM com um grupo de administradores para esse usuário.

Inscriva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a <https://portal.aws.amazon.com/billing/inscrição>.
2. Siga as instruções online.

Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWS é criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar [tarefas que exigem acesso de usuário-raiz](#).

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando <https://aws.amazon.com/e> escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em uma Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login [Console de gerenciamento da AWS](#) como proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

Para obter ajuda ao fazer login usando o usuário-raiz, consulte [Fazer login como usuário-raiz](#) no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

Para obter instruções, consulte [Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz \(console\) no Guia](#) do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

Para obter instruções, consulte [Habilitar o AWS IAM Identity Center](#) no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte [Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM](#) no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte [Como fazer login no portal de AWS acesso](#) no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

Para obter instruções, consulte [Criar um conjunto de permissões](#) no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de logon único ao grupo.

Para obter instruções, consulte [Adicionar grupos](#) no Guia do usuário do AWS IAM Identity Center .

Por exemplo, políticas AWS CloudHSM que você pode anexar ao seu grupo de usuários do IAM, consulte [Gerenciamento de identidade e acesso para AWS CloudHSM](#).

Crie uma nuvem privada virtual (VPC) para AWS CloudHSM

Você precisa de uma nuvem privada virtual (VPC) para seu cluster em. AWS CloudHSM Se você ainda não tem uma, siga as etapas neste tópico para criar uma VPC.

Note

Seguir essas etapas resultará na criação de sub-redes públicas e privadas.

Para criar uma VPC

1. Abra o console da Amazon VPC em <https://console.aws.amazon.com/vpc/>.
2. Na barra de navegação, use o seletor de região para escolher uma das [AWS regiões em que AWS CloudHSM há suporte no momento](#).
3. Selecione o botão Criar VPC.
4. Em Resources to create (Recursos a serem criados), escolha VPC and more (VPC e mais).
5. Em Nomear tag, digite um nome identificável como **CloudHSM**.
6. Para o bloco IPv6 CIDR, selecione o bloco IPv6 CIDR fornecido pela Amazon para usar a IPv6 conectividade para você HSMs e AWS alocue um bloco IPv6 CIDR para o seu cluster. Essa configuração é compatível com o Tipo de rede de pilha dupla. Mantenha a configuração padrão se você não precisar de IPv6 conectividade.

7. Deixe todas as outras opções com seus valores padrão.
8. Escolha Criar VPC.
9. Depois que a VPC for criada, selecione Exibir VPC para visualizar a VPC que você acabou de criar.

Crie um cluster em AWS CloudHSM

Um cluster é uma coleção de módulos individuais de segurança de hardware (HSMs). AWS CloudHSM sincroniza os HSMs em cada cluster para que funcionem como uma unidade lógica. AWS CloudHSM oferece dois tipos de HSMs: hsm1.medium e hsm2m.medium. Ao criar um cluster, você escolhe qual dos dois estará no cluster. Para obter detalhes sobre as diferenças entre cada tipo de HSM e modo de cluster, consulte [AWS CloudHSM modos de cluster](#).

Quando você cria um cluster, AWS CloudHSM cria um grupo de segurança para o cluster em seu nome. Esse grupo de segurança controla o acesso à rede HSMs no cluster. Ele permite conexões de entrada somente de instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) que estão no grupo de segurança. Por padrão, o security group não contém instâncias. Posteriormente, você [inicia uma instância do cliente](#) e [configura o grupo de segurança do cluster](#) para permitir a comunicação e as conexões com o HSM.

Considerações

- Veja a seguir algumas considerações ao criar um cluster no AWS CloudHSM:
 - Quando você cria um cluster, AWS CloudHSM cria uma [função vinculada ao serviço chamada AWSService RoleForCloud HSM](#). Se você AWS CloudHSM não conseguir criar a função ou se a função ainda não existir, talvez você não consiga criar um cluster. Para obter mais informações, consulte [Resolvendo falhas na criação de AWS CloudHSM clusters](#). Para obter mais informações sobre funções vinculadas ao serviço, consulte [Funções vinculadas a serviços para AWS CloudHSM](#).
 - Se você estiver usando o [endpoint de AWS CloudHSM pilha dupla](#) (ou seja, cloudhsmv2). `<region>.api.aws`), certifique-se de que suas políticas do IAM estejam atualizadas para serem processadas. IPv6 Para obter mais informações, consulte a [IPv6 seção Atualizar políticas do IAM para Segurança](#).

Você pode criar um cluster do [console do AWS CloudHSM](#), a [AWS Command Line Interface \(AWS CLI\)](#) ou a API do AWS CloudHSM .

Para obter detalhes sobre argumentos de cluster APIs, consulte [create-cluster](#) na Referência de AWS CLI Comandos.

Console

Para criar um cluster (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Na barra de navegação, use o seletor de região para selecionar uma das [regiões AWS onde o AWS CloudHSM é suportado atualmente](#).
3. Selecione Create cluster (Criar cluster).
4. Na seção Configuração de cluster, faça o seguinte:
 - a. Em VPC, selecione a VPC que você criou em [Crie uma nuvem privada virtual \(VPC\) para AWS CloudHSM](#).
 - b. Em Availability Zone(s), ao lado de cada Zona de disponibilidade, escolha a sub-rede privada que você criou.

Note

Mesmo que não AWS CloudHSM haja suporte em uma determinada zona de disponibilidade, o desempenho não deve ser afetado, pois balanceia AWS CloudHSM automaticamente a carga em todos os HSMs em um cluster. Consulte [AWS CloudHSM Regiões e endpoints](#) em Referência geral da AWS para ver o suporte da zona de disponibilidade para AWS CloudHSM.

- c. Para o tipo de HSM, selecione o tipo de HSM que pode ser criado em seu cluster junto com o modo desejado do cluster. Para ver quais tipos de HSM são compatíveis em cada região, consulte a [calculadora AWS CloudHSM de preços](#).

Important

Depois que o cluster for criado, o modo cluster não pode ser alterado. Para obter informações sobre qual tipo de cluster é adequado para seu caso de uso, consulte [AWS CloudHSM modos de cluster](#).

- d. Em Tipo de rede, escolha os protocolos de endereço IP para acessar seus HSMs. IPv4 limita a comunicação entre seu aplicativo e HSMs IPv4 apenas a. Essa é a opção

padrão. O Dual-Stack permite a comunicação e IPv4 a comunicação. IPv6 Para usar o dual-stack, adicione ambos IPv4 e às IPv6 CIDRs suas configurações de VPC e sub-rede. É difícil alterar o tipo de rede após a configuração inicial. Para modificá-lo, crie um backup do seu cluster existente e restaure um novo cluster com o tipo de rede desejado. Para obter mais informações, consulte [Criação de AWS CloudHSM clusters a partir de backups](#)

- e. Para a Origem do cluster, especifique se você deseja criar um cluster ou restaurar um de um backup existente.
 - Backups de clusters no modo não FIPS podem ser usados apenas para restaurar clusters que estão no modo não FIPS.
 - Backups de clusters no modo FIPS podem ser usados apenas para restaurar clusters que estão no modo FIPS.
5. Escolha Próximo.
6. Especifique por quanto tempo o serviço deve reter os backups.
 - Aceite o período de retenção padrão de 90 dias ou digite um novo valor entre 7 e 379 dias. O serviço excluirá automaticamente os backups presentes nesse cluster que sejam mais antigos do que o valor especificado. Você pode alterar esse valor depois. Para obter mais informações, consulte [Configurar a retenção de backup](#).
7. Escolha Próximo.
8. (Opcional) Digite uma chave de tag e um valor de tag opcional. Para adicionar mais de uma tag ao cluster, selecione Adicionar tag.
9. Escolha Review (Revisar).
10. Reveja sua configuração de cluster e escolha Create cluster (Criar cluster).

Se suas tentativas de criar um cluster falharem, isso pode estar relacionado a problemas com as funções AWS CloudHSM vinculadas ao serviço. Para obter ajuda para resolver essa falha, consulte [Resolvendo falhas na criação de AWS CloudHSM clusters](#).

AWS CLI

Para criar um cluster ([AWS CLI](#))

- Em um prompt de comando, execute o comando [create-cluster](#). Especifique o tipo de instância do HSM, o período de retenção de backup e a sub-rede IDs das sub-redes em

que você planeja criar. HSMs Use a sub-rede IDs das sub-redes privadas que você criou. Especifique apenas uma sub-rede por zona de disponibilidade.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
    --backup-retention-policy Type=DAYS,Value=<number of days> \
    --subnet-ids <subnet ID> \
    --mode <FIPS> \
    --network-type <IPV4>

{
  "Cluster": {
    "BackupPolicy": "DEFAULT",
    "BackupRetentionPolicy": {
      "Type": "DAYS",
      "Value": 90
    },
    "VpcId": "vpc-50ae0636",
    "SubnetMapping": {
      "us-west-2b": "subnet-49a1bc00",
      "us-west-2c": "subnet-6f950334",
      "us-west-2a": "subnet-fd54af9b"
    },
    "SecurityGroup": "sg-6cb2c216",
    "HsmType": "hsm2m.medium",
    "NetworkType": "IPV4",
    "Certificates": {},
    "State": "CREATE_IN_PROGRESS",
    "Hsms": [],
    "ClusterId": "cluster-igklspoyj5v",
    "ClusterMode": "FIPS",
    "CreateTimestamp": 1502423370.069
  }
}
```

Note

ClusterMode é um parâmetro obrigatório para todos os tipos de hsm, exceto hsm1.medium. --mode:

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
    --backup-retention-policy Type=DAYS,Value=<number of days> \
    --subnet-ids <subnet ID> \
```

```
--mode NON_FIPS
```

Se suas tentativas de criar um cluster falharem, isso pode estar relacionado a problemas com as funções AWS CloudHSM vinculadas ao serviço. Para obter ajuda para resolver essa falha, consulte [Resolvendo falhas na criação de AWS CloudHSM clusters](#).

AWS CloudHSM API

Para criar um cluster (AWS CloudHSM API)

- Envie uma solicitação [CreateCluster](#). Especifique o tipo de instância do HSM, a política de retenção de backup e a sub-rede IDs das sub-redes em que você planeja criar. HSMs Use a sub-rede IDs das sub-redes privadas que você criou. Especifique apenas uma sub-rede por zona de disponibilidade.

Se suas tentativas de criar um cluster falharem, isso pode estar relacionado a problemas com as funções AWS CloudHSM vinculadas ao serviço. Para obter ajuda para resolver essa falha, consulte [Resolvendo falhas na criação de AWS CloudHSM clusters](#).

Analise o grupo de segurança do seu cluster no AWS CloudHSM

Quando você cria um cluster ou adiciona um HSM a um cluster, AWS CloudHSM cria um grupo de segurança com o nome, `cloudhsm-cluster-<clusterID>-sg` caso ainda não exista. Esse grupo de segurança contém uma regra de TCP pré-configurada que permite a comunicação de entrada e de saída no grupo de segurança do cluster por meio das portas 2223-2225. Esse SG permite que suas instâncias do EC2 usem sua VPC para se comunicar em seu cluster. HSMs

Warning

- Não exclua ou modifique a regra de TCP pré-configurada, que é preenchida no grupo de segurança do cluster. Essa regra pode evitar problemas de conectividade e acesso não autorizado ao seu HSMs.
- O grupo de segurança do cluster impede o acesso não autorizado ao seu HSMs. Qualquer pessoa que possa acessar instâncias no grupo de segurança pode acessar seu HSMs. A maioria das operações exige que um usuário faça login no HSM. No entanto, é possível zerar HSMs sem autenticação, o que destrói o material da chave, os certificados e outros

dados. Se isso acontecer, os dados criados ou modificados após o backup mais recente serão perdidos e não poderão ser recuperados. Para impedir o acesso não autorizado, certifique-se de que apenas os administradores confiáveis podem modificar ou acessar as instâncias no grupo de segurança padrão.

- Os clusters hsm2m.medium introduzem o recurso mTLS para impedir que usuários não autorizados se conectem ao cluster. Usuários não autorizados precisarão de credenciais mTLS válidas para se conectarem com êxito ao cluster antes de tentarem a zeragem.

Na próxima etapa, você pode [iniciar uma instância do Amazon EC2](#) e conectá-la à sua HSMs [anexando o grupo de segurança do cluster a](#) ela.

Inicie uma instância cliente do Amazon EC2 para interagir com AWS CloudHSM

Para interagir e gerenciar seu AWS CloudHSM cluster e suas instâncias de HSM, você deve ser capaz de se comunicar com as interfaces de rede elástica de seus HSMs. A maneira mais fácil de fazer isso é usar uma instância do EC2 na mesma VPC do cluster. Use também os recursos da AWS a seguir para se conectar a seu cluster:

- [Emparelhamento do Amazon VPC](#)
- [Direct Connect](#)
- [Conexões VPN](#)

Note

Este guia fornece um exemplo simplificado de como conectar uma instância do EC2 ao seu AWS CloudHSM cluster. Para obter as práticas recomendadas sobre configurações de rede seguras, consulte [Acesso seguro ao cluster](#).

A AWS CloudHSM documentação normalmente presume que você está usando uma instância do EC2 na mesma VPC e zona de disponibilidade (AZ) em que você cria seu cluster.

Para criar uma instância do EC2

1. Abra o painel do EC2 em. <https://console.aws.amazon.com/ec2/>
2. Selecione Iniciar instância. No menu suspenso, selecione Iniciar instância.
3. No campo Nome, insira um nome para a instância EC2.
4. Na seção Aplicativos e imagens OS (Amazon Machine Image), escolha uma imagem de máquina da Amazon (AMI) que corresponda a uma plataforma compatível com o CloudHSM. Para obter mais informações, consulte [AWS CloudHSM Plataformas compatíveis com o Client SDK 5](#).
5. Na seção Tipo de instância, selecione o tipo de instância.
6. Na seção Par de chaves, use um par de chaves existente ou selecione Criar novo par de chaves e conclua as seguintes etapas:
 - a. Em Nome do par de chaves, insira um nome para o novo par de chaves.
 - b. Em Par de chaves, escolha um par de chaves.
 - c. Para Formato de arquivo de chave privada, escolha o formato no qual salvar a chave privada.
 - d. Selecione Criar par de chaves.
 - e. Baixe e salve o arquivo de chave privada.

Important

Esta é sua única oportunidade de salvar o arquivo de chave privada. Baixe e armazene o arquivo em um lugar seguro. Você precisa fornecer o nome do par de chaves ao iniciar uma instância. Além disso, você deve fornecer a chave privada correspondente sempre que se conectar à instância e escolher o par de chaves que criou ao instalar.

7. Em Configurações de rede, selecione Editar.
8. Em VPC, escolha a VPC criada anteriormente para o cluster.
9. Em Subnet (Sub-rede), selecione a sub-rede pública criada anteriormente para a VPC.
10. Para Auto-assign Public IP (Atribuir IP público automaticamente), selecione Permitir.
11. Para atribuição automática de IPv6 IP, escolha Habilitar para usar a conectividade IPv6 com seus clusters e o Dual-Stack. NetworkType Se você habilitar essa opção, atualize as regras do

- grupo de segurança, as tabelas de rotas da VPC e da sub-rede e a rede da sua instância do Amazon EC2 ACLs para permitir o tráfego de IPv6 saída da instância para o. HSMs
12. Escolha Select an existing security group (Selecionar um grupo de segurança existente).
 13. Em Grupos de segurança comuns, selecione o grupo de segurança padrão no menu suspenso.
 14. Em Configurar armazenamento, use os menus suspensos para escolher uma configuração de armazenamento.
 15. Na janela Resumo, selecione Iniciar instância.

 Note

A conclusão dessa etapa iniciará o processo de criação de sua instância do EC2.

Para obter mais informações sobre como criar um cliente do para Linux, consulte [Conceitos básicos das instâncias do Linux do Amazon EC2](#). Para obter informações sobre como conectar-se ao cliente em execução, consulte os seguintes tópicos:

- [Conexão à sua instância do Linux utilizando SSH](#)
- [Conexão com a instância do Linux no Windows utilizando PuTTY](#)

O guia do usuário do Amazon EC2 contém instruções detalhadas para configurar e usar suas instâncias do Amazon EC2. A lista a seguir fornece uma visão geral da documentação disponível para clientes do Linux e do Windows Amazon EC2:

- Para criar um cliente do Amazon EC2 do Linux, consulte [Conceitos básicos das instâncias do Linux do Amazon EC2](#).

Para obter informações sobre como conectar-se ao cliente em execução, consulte os seguintes tópicos:

- [Conexão à sua instância do Linux utilizando SSH](#)
- [Conexão com a instância do Linux no Windows utilizando PuTTY](#)
- Para criar um cliente do Amazon EC2 do Windows, consulte [Conceitos básicos de instâncias do Windows Amazon EC2](#). Para obter mais informações sobre como conectar-se a seu cliente do Windows, consulte [Conexão à sua instância do Windows](#).

 Note

Sua instância do EC2 pode executar todos os AWS CLI comandos contidos neste guia. Se a AWS CLI não estiver instalada, você poderá fazer download da [AWS Command Line Interface](#). Se você estiver usando o Windows, poderá fazer download e executar um instalador do Windows de 64 bits ou de 32 bits. Se estiver usando o Linux ou o macOS, você poderá instalar a CLI usando o pip.

Configure os grupos de segurança da instância cliente Amazon EC2 para AWS CloudHSM

Ao iniciar uma instância do Amazon EC2 para seu cluster em AWS CloudHSM, você a associou a um grupo de segurança padrão da Amazon VPC. Este tópico explica como associar o grupo de segurança do cluster à instância do EC2. Essa associação permite que o AWS CloudHSM cliente em execução na sua instância do EC2 se comunique com seus HSMs. Para conectar sua instância do EC2 ao seu AWS CloudHSM cluster, você deve configurar adequadamente o grupo de segurança padrão da VPC e associar o grupo de segurança do cluster à instância.

Siga as etapas a seguir para concluir as alterações na configuração.

Tópicos

- [Etapa 1. Modificar o grupo de segurança padrão](#)
- [Etapa 2. Conecte a instância do Amazon EC2 ao cluster AWS CloudHSM](#)

Etapa 1. Modificar o grupo de segurança padrão

Você precisa modificar o grupo de segurança padrão para permitir a conexão SSH ou RDP para que seja possível fazer download e instalar o software cliente e interagir com o HSM.

Para modificar o grupo de segurança padrão

1. Abra o painel do EC2 em: <https://console.aws.amazon.com/ec2/>
2. Selecione Instâncias (em execução) e, em seguida, marque a caixa de seleção ao lado da instância EC2 na qual você deseja instalar o AWS CloudHSM cliente.
3. Na guia Segurança, escolha o grupo de segurança chamado Padrão.

4. Na parte superior da página, escolha Actions (Ações) e depois Edit inbound rules (Editar regras de entrada).
5. Selecione Adicionar regra.
6. Em Type (Tipo), siga um destes procedimentos:
 - Para uma instância do Amazon EC2 do Windows Server, escolha RDP. A porta 3389 é preenchida automaticamente.
 - Para uma instância do Amazon EC2 do Linux, escolha SSH. O intervalo de portas 22 é preenchido automaticamente.
7. Para qualquer uma das opções, defina Fonte como Meu IP para permitir que você se comunique com sua instância do Amazon EC2.

 Important

Não especifique 0.0.0.0/0 como o intervalo CIDR para evitar que qualquer pessoa acesse sua instância.

8. Escolha Save (Salvar).

Etapa 2. Conecte a instância do Amazon EC2 ao cluster AWS CloudHSM

Você deve anexar o grupo de segurança do cluster à instância do EC2 para que a instância do EC2 possa se comunicar com HSMs seu cluster. O grupo de segurança do cluster contém uma regra pré-configurada que permite comunicação de entrada pelas portas 2223-2225.

Para conectar a instância do EC2 ao cluster AWS CloudHSM

1. Abra o painel do EC2 em <https://console.aws.amazon.com/ec2/>
2. Selecione Instâncias (em execução) e, em seguida, marque a caixa de seleção da instância EC2 na qual você deseja instalar o AWS CloudHSM cliente.
3. Na parte superior da página, escolha Ações, Segurança e depois Alterar grupos de segurança.
4. Selecione o grupo de segurança com o nome do grupo que corresponde ao ID do cluster, como `cloudhsm-cluster-<clusterID>-sg`.
5. Escolha Adicionar grupos de segurança.
6. Selecione Save (Salvar).

 Note

Você pode atribuir no máximo cinco grupos de segurança a uma instância do Amazon EC2. Se tiver atingido o limite máximo, você deverá modificar o grupo de segurança padrão da instância do Amazon EC2 e o grupo de segurança do cluster:

No grupo de segurança padrão, faça o seguinte:

- Adicione uma regra de entrada para permitir tráfego usando o protocolo TCP por meio das portas 2223-2225 a partir do grupo de segurança do cluster.

No grupo de segurança do cluster, faça o seguinte:

- Adicione uma regra de entrada para permitir tráfego usando o protocolo TCP por meio das portas 2223-2225 do grupo de segurança padrão.

Crie um HSM em AWS CloudHSM

Depois de criar um cluster no AWS CloudHSM, você pode criar um módulo de segurança de hardware (HSM). No entanto, antes de criar um HSM no seu cluster, o cluster deve estar em um estado não inicializado. Para determinar o estado do cluster, visualize a [página de clusters no AWS CloudHSM console](#), use o AWS CLI para executar o [describe-clusters](#) comando ou envie uma [DescribeClusters](#) solicitação na AWS CloudHSM API. Você pode criar um HSM do [console do AWS CloudHSM](#), a [AWS CLI](#) ou a API do AWS CloudHSM .

 Important

Crie apenas um HSM enquanto seu cluster estiver no estado não inicializado.

Console

Para criar um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione o botão de seleção ao lado do ID do cluster para o qual deseja criar um HSM.
3. Selecione Ações. No menu suspenso, escolha Iniciar.

4. Escolha uma zona de disponibilidade (AZ) para o HSM que está sendo criado.
5. Escolha Criar.

Depois de criar um cluster e um HSM, opcionalmente, é possível [verificar a identidade do HSM](#) ou ir diretamente para [Inicializar o cluster](#).

AWS CLI

Para criar um HSM ([AWS CLI](#))

- Em um prompt de comando, execute o comando [create-hsm](#). Especifique o ID do cluster criado anteriormente e uma zona de disponibilidade para o HSM. Especifique a zona de disponibilidade no formato us-west-2a, us-west-2b etc.

```
$ aws cloudhsmv2 create-hsm --cluster-id <cluster ID> --availability-  
zone <Availability Zone>  
  
{  
  "Hsm": {  
    "HsmId": "hsm-ted36yp5b2x",  
    "EniIp": "10.0.1.12",  
    "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",  
    "AvailabilityZone": "us-west-2a",  
    "ClusterId": "cluster-igklspoyj5v",  
    "EniId": "eni-5d7ade72",  
    "SubnetId": "subnet-fd54af9b",  
    "State": "CREATE_IN_PROGRESS"  
  }  
}
```

Depois de criar um cluster e um HSM, opcionalmente, é possível [verificar a identidade do HSM](#) ou ir diretamente para [Inicializar o cluster](#).

AWS CloudHSM API

Para criar um HSM (AWS CloudHSM API)

- Envie uma solicitação [CreateHsm](#). Especifique o ID do cluster criado anteriormente e uma zona de disponibilidade para o HSM.

Depois de criar um cluster e um HSM, opcionalmente, é possível [verificar a identidade do HSM](#) ou ir diretamente para [Inicializar o cluster](#).

Verifique a identidade e a autenticidade do HSM do seu cluster em AWS CloudHSM (opcional)

Para inicializar seu cluster AWS CloudHSM, você assina uma solicitação de assinatura de certificado (CSR) gerada pelo primeiro módulo de segurança de hardware (HSM) do cluster. Antes de fazer isso, você pode verificar a identidade e a autenticidade do HSM.

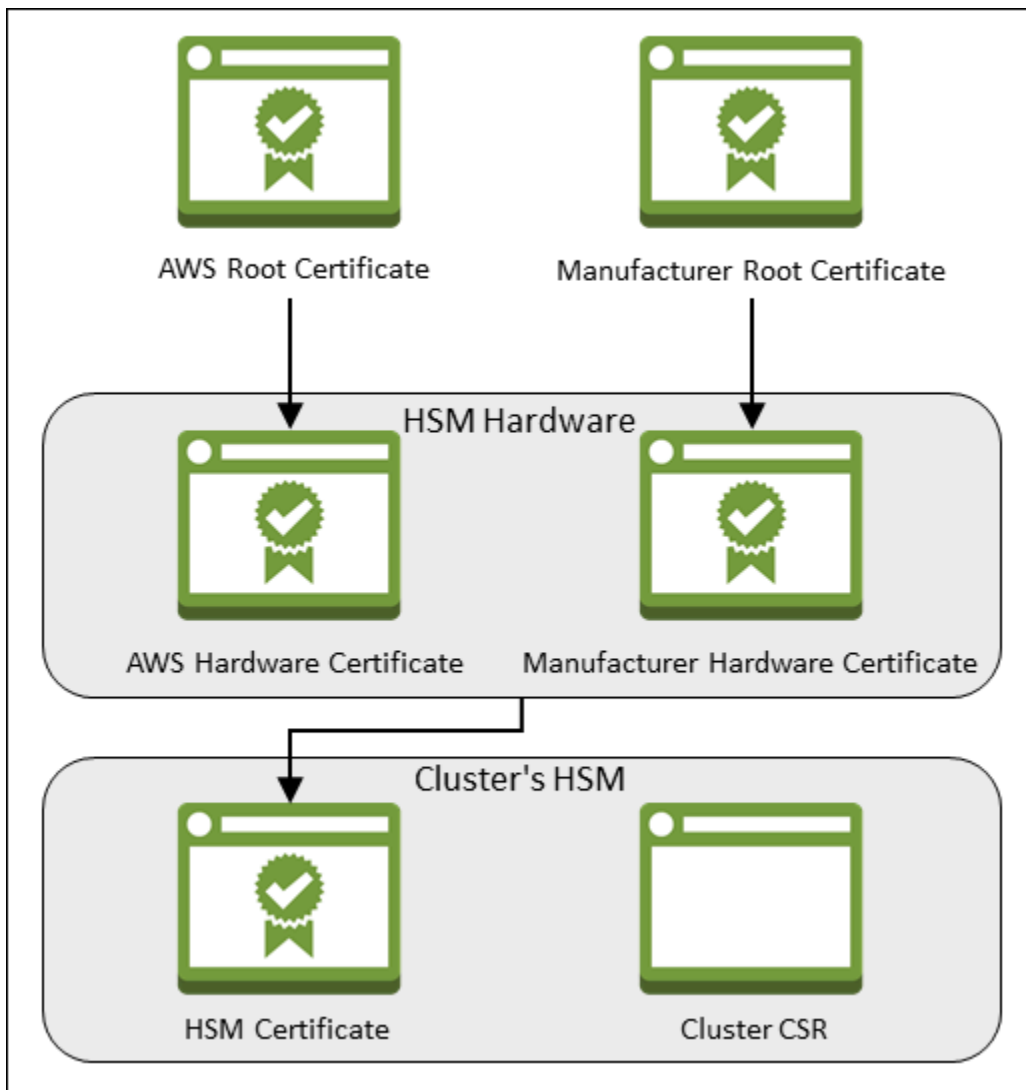
Note

Este processo é opcional. No entanto, ele funciona apenas até que um cluster seja inicializado. Depois que o cluster for inicializado, você não poderá usar esse processo para obter os certificados ou verificar os HSMs.

Para verificar a identidade do primeiro HSM do cluster, execute as seguintes etapas:

1. [Obter os certificados e CSR](#) : nesta etapa, são obtidos três certificados e uma CSR no HSM. Você também recebe dois certificados raiz, um do fabricante do hardware do HSM AWS CloudHSM e outro.
2. [Verifique as cadeias de certificados](#) — Nesta etapa, você constrói duas cadeias de certificados, uma para o certificado AWS CloudHSM raiz e outra para o certificado raiz do fabricante. Em seguida, você verifica o certificado do HSM com essas cadeias de certificados para determinar isso AWS CloudHSM e o fabricante do hardware atesta a identidade e a autenticidade do HSM.
3. [Comparar chaves públicas](#): nesta etapa, as chaves públicas são extraídas e comparadas com as chaves públicas no certificado do HSM e na CSR do cluster, para garantir que são iguais. Isso deve fornecer a confiança de que o CSR foi gerado por um HSM autêntico e confiável.

O diagrama a seguir mostra a CSR, os certificados e a relação entre eles. Cada certificado é definido pela lista subsequente.



AWS Certificado raiz

Esse é AWS CloudHSM o certificado raiz.

Certificado raiz do fabricante

Este é o certificado raiz do fabricante de hardware.

AWS Certificado de hardware

AWS CloudHSM criou esse certificado quando o hardware do HSM foi adicionado à frota. Esse certificado afirma que AWS CloudHSM é proprietário do hardware.

Certificado de hardware do fabricante

O fabricante de hardware do HSM criou esse certificado quando o hardware do HSM foi fabricado. Esse certificado garante que o fabricante criou o hardware.

Certificado HSM

O certificado do HSM é gerado pelo hardware validado pelo FIPS quando você cria o primeiro HSM no cluster. Esse certificado garante que o hardware do HSM criou o HSM.

CSR do cluster

O primeiro HSM cria a CSR do cluster. Ao [assinar a CSR do cluster](#), você reivindica o cluster. Em seguida, você pode usar a CSR assinada para [inicializar o cluster](#).

Etapa 1. Obter certificados do HSM

Para verificar a identidade e a autenticidade do HSM, começar obtendo um CSR e cinco certificados. Você recebe três dos certificados do HSM, o que pode ser feito com o [AWS CloudHSM console](#), o [AWS Command Line Interface \(AWS CLI\)](#) ou a AWS CloudHSM API.

Console

Para obter a CSR e certificados de HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione o botão de seleção ao lado do ID do cluster com o HSM que deseja verificar.
3. Selecione Ações. No menu suspenso, escolha Iniciar.
4. Se você não concluiu a [etapa anterior](#) para criar um HSM, escolha uma Zona de Disponibilidade (AZ) para o HSM que você está criando. Em seguida, selecione Criar.
5. Quando os certificados e a CSR estiverem prontos, você verá links para fazer o download.

Certificate signing request

To initialize the cluster, you must download a certificate signing request (CSR) and then [sign it](#) .

 **Cluster CSR**

Cluster verification certificate

Optionally, you may wish to download the HSM certificate below which generated this Cluster CSR and [verify its authenticity](#) .

 **HSM certificate**

6. Escolha cada link para fazer download e salvar a CSR e seus certificados. Para simplificar as etapas subsequentes, salve todos os arquivos no mesmo diretório e use os nomes de arquivo padrão.

AWS CLI

Para obter a CSR e os certificados HSM ([AWS CLI](#))

- No prompt de comando, execute o comando [describe-clusters](#) quatro vezes, extraindo o CSR e certificados diferentes cada vez e salvando-os em arquivos.

- a. Emita o seguinte comando para extrair a CSR do cluster. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                     --output text \
                                     --query
'Clusters[].Certificates.ClusterCsr' \
                                     > <cluster ID>_ClusterCsr.csr
```

- b. Emita o seguinte comando para extrair o certificado do HSM. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                     --output text \
                                     --query
'Clusters[].Certificates.HsmCertificate' \
                                     > <cluster ID>_HsmCertificate.crt
```

- c. Execute o comando a seguir para extrair o certificado AWS de hardware. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                     --output text \
                                     --query
'Clusters[].Certificates.AwsHardwareCertificate' \
                                     > <cluster ID>_AwsHardwareCertificate.crt
```

- d. Emita o seguinte comando para extrair o certificado de hardware do fabricante. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                     --output text \
                                     --query
'Clusters[].Certificates.ManufacturerHardwareCertificate' \
                                     > <cluster ID>_ManufacturerHardwareCertificate.crt
```

AWS CloudHSM API

Para obter os certificados CSR e HSM (API)AWS CloudHSM

- Envie uma solicitação [DescribeClusters](#) e, em seguida, extraia e salve a CSR e os certificados da resposta.

Etapa 2. Obter certificados raiz

Siga estas etapas para obter os certificados raiz AWS CloudHSM e o fabricante. Salve os arquivos de certificado raiz no diretório que contém os arquivos CSR e de certificado da HSM.

Para obter os certificados raiz AWS CloudHSM e do fabricante

1. Baixe o certificado AWS CloudHSM raiz: [AWS_CloudHSM_Root-G1.zip](#)
2. Faça download do certificado raiz do fabricante certo para seu tipo de HSM:
 - Certificado raiz do fabricante do hsm1.medium: [liquid_security_certificate.zip](#)
 - Certificado raiz do fabricante do hsm2.medium: [liquid_security_certificate.zip](#)

Note

Para fazer download de cada certificado da respectiva página inicial, use os links a seguir:

- Página inicial do [certificado raiz do fabricante](#) da instância hsm1.medium
- Página inicial do [certificado raiz do fabricante](#) da instância hsm2m.medium

Talvez seja necessário clicar no link de Fazer download de certificado e escolher Salvar Link como... a seguir, para salvar o arquivo do certificado.

3. Depois de fazer o download dos arquivos, extraia (descompacte) seu conteúdo.

Etapa 3. Verificar cadeias de certificados

Nesta etapa, você constrói duas cadeias de certificados, uma para o certificado AWS CloudHSM raiz e outra para o certificado raiz do fabricante. Em seguida, use OpenSSL para verificar o certificado de HSM com cada cadeia de certificado.

Para criar as cadeias de certificados, abra um shell do Linux. Você precisa do OpenSSL, que está disponível na maioria dos shells do Linux e precisa do [certificado raiz](#) e dos [arquivos de certificado do HSM](#) que foram transferidos por download. No entanto, você não precisa do AWS CLI para esta etapa e o shell não precisa estar associado à sua AWS conta.

Para verificar o certificado HSM com o certificado AWS CloudHSM raiz

1. Navegue até o diretório onde você salvou o [certificado raiz](#) e os [arquivos de certificado do HSM](#) que foram transferidos por download. Os comandos a seguir assumem que todos os certificados estejam no diretório atual e usam os nomes de arquivo padrão.

Use o comando a seguir para criar uma cadeia de certificados que inclua o certificado de AWS hardware e o certificado AWS CloudHSM raiz, nessa ordem. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ cat <cluster ID>_AwsHardwareCertificate.crt \  
    AWS_CloudHSM_Root-G1.crt \  
    > <cluster ID>_AWS_chain.crt
```

2. Use o comando OpenSSL a seguir para verificar o certificado de HSM com a cadeia de certificação da AWS. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ openssl verify -CAfile <cluster ID>_AWS_chain.crt <cluster ID>_HsmCertificate.crt  
<cluster ID>_HsmCertificate.crt: OK
```

Para verificar o certificado de HSM com o certificado raiz do fabricante

1. Use o comando a seguir para criar uma cadeia de certificado que inclua o certificado de hardware do fabricante e o certificado raiz do fabricante, nesta ordem. *<cluster ID>* Substitua pelo ID do cluster que você criou anteriormente.

```
$ cat <cluster ID>_ManufacturerHardwareCertificate.crt \  
    liquid_security_certificate.crt \  
    > <cluster ID>_ManufacturerChain.crt
```

```
> <cluster ID>_manufacturer_chain.crt
```

2. Use o comando OpenSSL a seguir para verificar o certificado do HSM com a cadeia de certificados do fabricante. <cluster ID> Substitua pelo ID do cluster que você criou anteriormente.

```
$ openssl verify -CAfile <cluster ID>_manufacturer_chain.crt <cluster ID>_HsmCertificate.crt
<cluster ID>_HsmCertificate.crt: OK
```

Etapa 4: Extrair e comparar chaves públicas

Use OpenSSL para extrair e comparar chaves públicas no certificado do HSM e na CSR do cluster, a fim de garantir que são iguais.

Para comparar as chaves públicas, use o shell do Linux. Você precisa do OpenSSL, que está disponível na maioria dos shells Linux, mas não é necessário AWS CLI para esta etapa. O shell não precisa estar associado à sua AWS conta.

Para extrair e comparar as chaves públicas

1. Use o comando a seguir para extrair a chave pública do certificado de HSM.

```
$ openssl x509 -in <cluster ID>_HsmCertificate.crt -pubkey -noout > <cluster ID>_HsmCertificate.pub
```

2. Use o comando a seguir para extrair a chave pública da CSR do cluster.

```
$ openssl req -in <cluster ID>_ClusterCsr.csr -pubkey -noout > <cluster ID>_ClusterCsr.pub
```

3. Use o comando a seguir para comparar as chaves públicas. Se as chaves públicas forem idênticas, o comando a seguir não produzirá resultado.

```
$ diff <cluster ID>_HsmCertificate.pub <cluster ID>_ClusterCsr.pub
```

Depois de verificar a identidade e a autenticidade do HSM, vá para [Inicializar o cluster](#).

Inicialize o cluster em AWS CloudHSM

Depois de criar seu cluster e adicionar seu módulo de segurança de hardware (HSM) AWS CloudHSM, você pode inicializar o cluster. Conclua as etapas dos tópicos a seguir para inicializar o cluster do .

Note

Antes de inicializar o cluster, revise o processo pelo qual você pode [verificar a identidade e a autenticidade do. HSMs](#) Esse processo é opcional e funciona apenas até que um cluster seja inicializado. Depois que o cluster for inicializado, você não poderá usar esse processo para obter seus certificados ou verificar o. HSMs

Tópicos

- [Visão geral do](#)
- [Etapa 1. Obter a Solicitação de assinatura de certificado \(CSR\) do cluster](#)
- [Etapa 2. Crie uma chave privada para sua CA raiz](#)
- [Etapa 3. Assinar a CSR](#)
- [Etapa 4: Inicializar o cluster](#)

Visão geral do

O processo de inicialização do cluster estabelece sua propriedade e controle sobre o cluster e sobre você HSMs por meio de um sistema de autenticação baseado em certificado. Esse processo prova criptograficamente que você é o único proprietário do HSMs em seu cluster e cria a base de confiança que será necessária para todas as conexões futuras com o seu. HSMs

Esta página mostrará como fazer o seguinte:

- Recupere a solicitação de assinatura de certificado (CSR) do seu cluster.
- Gere e use a (s) chave (s) privada (s) para criar um certificado raiz autoassinado ou uma cadeia de certificados.
- Assine a CSR do seu cluster para produzir um certificado HSM assinado.
- Inicialize seu cluster usando o certificado HSM assinado e o certificado autoassinado ou a cadeia de certificados.

Quando estiver pronto para começar, vá para [Etapa 1. Obter a Solicitação de assinatura de certificado \(CSR\) do cluster](#).

Etapa 1. Obter a Solicitação de assinatura de certificado (CSR) do cluster

Antes de inicializar o cluster, é necessário baixar e assinar uma solicitação de assinatura de certificado (CSR) gerada pelo primeiro HSM do cluster. Se tiver seguido as etapas para [verificar a identidade do HSM do cluster](#), você já deverá ter a CSR e poderá assiná-la. Caso contrário, obtenha a CSR agora usando o [AWS CloudHSM console](#), o [AWS Command Line Interface \(AWS CLI\)](#) ou a AWS CloudHSM API.

Console

Para obter a CSR (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione o botão de seleção ao lado do ID do cluster com o HSM que deseja verificar.
3. Selecione Ações. No menu suspenso, escolha Iniciar.
4. Se você não concluiu a [etapa anterior](#) para criar um HSM, escolha uma Zona de Disponibilidade (AZ) para o HSM que você está criando. Em seguida, selecione Criar.
5. Quando a CSR estiver pronta, você verá um link para fazer o download.

Certificate signing request

To initialize the cluster, you must download a certificate signing request (CSR) and then [sign it](#) .

 **Cluster CSR**

Cluster verification certificate

Optionally, you may wish to download the HSM certificate below which generated this Cluster CSR and [verify its authenticity](#) .

 **HSM certificate**

6. Selecione Cluster CSR para fazer o download e salvar a CSR.

AWS CLI

Para obter a CSR ([AWS CLI](#))

- No prompt de comando, execute o seguinte comando [describe-clusters](#), que extrai a CSR e a salva em um arquivo. **<cluster ID>** Substitua pelo ID do cluster que você [criou anteriormente](#).

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \  
--output text \  

```

```
\
--query 'Clusters[].Certificates.ClusterCsr'
> <cluster ID>_ClusterCsr.csr
```

AWS CloudHSM API

Para obter o CSR (AWS CloudHSM API)

1. Envie uma solicitação [DescribeClusters](#).
2. Extraia e salve a CSR da resposta.

Etapa 2. Crie uma chave privada para sua CA raiz

Note

Para um cluster de produção, a chave deve ser criada com segurança usando uma fonte confiável de aleatoriedade. Recomendamos que você use um HSM fora do local e off-line ou o equivalente. Armazene a chave com segurança. A chave estabelece a identidade do cluster e seu controle exclusivo sobre o HSMs que ele contém.

Durante a fase de desenvolvimento e teste, você pode usar qualquer ferramenta conveniente (como o OpenSSL) para criar e assinar o certificado do cluster. O exemplo a seguir mostra como criar uma chave. Após usar a chave para criar um certificado autoassinado (veja abaixo), você deve armazená-la com segurança. Para entrar na sua AWS CloudHSM instância, o certificado precisa estar presente, mas a chave privada não.

A tabela abaixo descreve os algoritmos, tamanhos de chave e curvas compatíveis para a geração de certificados.

Algoritmos	Tamanho/Curvas
RSA 5 PKCSv1.	2048, 3072, 4096
RSA-PSS	2048, 3072, 4096
ECDSA	prime256v1, secp384r1, secp521r1

Algoritmos	Tamanho/Curvas
Resumo	SHA-224, SHA-256, SHA-384 e SHA-512

Use o comando de exemplo a seguir para criar uma chave privada para sua CA raiz autoassinada.

```
$ openssl genrsa -aes256 -out customerRootCA.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for customerRootCA.key:
Verifying - Enter pass phrase for customerRootCA.key:
```

Etapa 3. Assinar a CSR

Nas etapas anteriores, você recuperou a CSR do seu cluster e criou uma chave privada para sua CA raiz. Nesta etapa, você usará sua chave privada para gerar um certificado de assinatura para assinar a CSR do seu cluster. Os tópicos abaixo guiarão você pelo processo de criação de um único certificado autoassinado, ou uma cadeia de certificados, usando o OpenSSL. Você não precisa do AWS CLI para esta etapa e o shell não precisa estar associado à sua AWS conta.

Important

Para inicializar seu cluster, sua âncora de confiança deve estar em conformidade com a [RFC 5280](#) e atender aos seguintes requisitos:

- Se estiver usando extensões X509v3, a extensão X509v3 Basic Constraints deve estar presente.
- A âncora de confiança deve ser um certificado autoassinado.
- Os valores de extensão não devem entrar em conflito uns com os outros.

Escolha uma das seguintes abordagens para assinar a CSR do seu cluster:

Escolha sua abordagem de certificação

Você deve escolher uma das duas abordagens a seguir. Não conclua as duas abordagens.

Opção A: certificado autoassinado único

Crie um único certificado raiz autoassinado para assinar a CSR do seu cluster. Esse é o método mais simples e direto de estabelecer confiança.

Recomendado para:

- Ambientes em que uma PKI externa não é necessária
- Ambientes de teste e desenvolvimento em que a simplicidade é preferida

Vá para: [Crie um único certificado autoassinado](#)

Opção B: cadeia de certificados com CA intermediária

Crie uma cadeia de certificados usando uma autoridade de certificação intermediária. Uma cadeia de certificados intermediária fornece segurança, escalabilidade e flexibilidade aprimoradas, permitindo que as Autoridades de Certificação raiz (CAs) permaneçam off-line enquanto delegam a emissão do certificado ao intermediário CAs, reduzindo assim o risco de comprometer a CA raiz.

Recomendado para:

- Ambientes em que uma PKI externa é necessária
- Integração com a Autoridade de Certificação Privada (PCA) da AWS

Exemplo de integração com o AWS PCA: você pode usar a Autoridade de Certificação AWS Privada para criar e gerenciar seus certificados CA intermediários. Isso fornece gerenciamento automatizado do ciclo de vida do certificado, incluindo renovação e revogação, enquanto mantém os benefícios de segurança de manter sua CA raiz offline. Para obter mais informações sobre o AWS PCA, consulte o [Guia do usuário da Autoridade de Certificação Privada da AWS](#).

Vá para: [Crie uma cadeia de autoridade de certificação intermediária \(ICA\)](#)

Crie um único certificado autoassinado

O hardware confiável que você usa para criar a chave privada para seu cluster de produção também deve fornecer uma ferramenta de software para gerar um certificado autoassinado usando essa chave. O exemplo a seguir usa o OpenSSL e a chave privada que você criou na etapa anterior para criar um certificado de assinatura de CA raiz autoassinado. O certificado é válido por 10 anos (3652 dias). Leia as instruções na tela e siga os avisos.


```
$ openssl req -new -x509 -days 3652 -key customerRootCA.key -out customerRootCA.crt
Enter pass phrase for customerRootCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Este comando cria um arquivo de certificado nomeado `customerRootCA.crt`. Coloque esse certificado em cada host a partir do qual você se conectará ao seu AWS CloudHSM cluster. Se você der um nome diferente ao arquivo ou armazená-lo em um caminho diferente da raiz do host, edite o arquivo de configuração do cliente adequadamente. Use o certificado e a chave privada que você acabou de criar para assinar a solicitação de assinatura de certificado (CSR) do cluster na próxima etapa.

Assine o CSR do cluster com sua CA raiz autoassinada

O hardware confiável que você usa para criar a chave privada do cluster de produção também deve fornecer uma ferramenta para assinar a CSR por meio dessa chave. O exemplo a seguir usa o OpenSSL para assinar a CSR do cluster. O comando de exemplo abaixo assina o CSR com o autoassinado `customerRootCA.crt`

```
$ openssl x509 -req -days 3652 -in <cluster ID>_ClusterCsr.csr \
  -CA <customerRootCA>.crt \
  -CAkey <customerRootCA>.key \
  -CAcreateserial \
  -out <cluster ID>_CustomerHsmCertificate.crt
Signature ok
subject=/C=US/ST=CA/O=Cavium/OU=N3FIPS/L=SanJose/CN=HSM:<HSM
  identifier>:PARTN:<partition number>, for FIPS mode
Getting CA Private Key
Enter pass phrase for <customerRootCA>.key:
```

Este comando cria um arquivo chamado `<cluster ID>_CustomerHsmCertificate.crt`. Use este arquivo como o certificado assinado ao inicializar o cluster.

Verifique o certificado assinado em relação à CA raiz (opcional):

```
$ openssl verify -purpose sslserver -CAfile customerRootCA.crt <cluster ID>_CustomerHsmCertificate.crt
<cluster ID>_CustomerHsmCertificate.crt: OK
```

Depois de produzir o certificado HSM assinado com sua CA raiz autoassinada, acesse. [Etapa 4: Inicializar o cluster](#)

Crie uma cadeia de autoridade de certificação intermediária (ICA)

Os exemplos a seguir ajudarão você a criar uma cadeia de certificados de comprimento 2, consistindo em uma Autoridade Certificadora (CA) raiz e uma CA intermediária. Primeiro, você criará um certificado de CA raiz autoassinado e, em seguida, gerará uma CA intermediária assinada pela CA raiz. Por fim, você usará a CA intermediária para assinar a CSR do seu cluster, criando uma cadeia de confiança completa do seu certificado HSM até a CA raiz. Essa abordagem fornece segurança aprimorada ao manter a CA raiz off-line enquanto usa a CA intermediária para operações de day-to-day certificado.

Important

Para inicializar seu cluster com uma cadeia de certificados, sua cadeia deve atender aos seguintes requisitos:

- A cadeia deve ser solicitada, começando com a CA intermediária que assina a CSR do cluster. Nessa ordem, o primeiro ICA deve ter um emissor que corresponda ao assunto do próximo ICA na cadeia, e assim por diante.
- Somente a CA raiz deve ser autoassinada, o que significa que o emissor e o assunto devem ser idênticos.
- A cadeia deve consistir em no máximo 4 certificados (incluindo a CA raiz no final) e o tamanho total da cadeia não deve exceder 16 kb (kilobytes).
- Todas as autoridades de certificação (CAs) devem estar em conformidade com as diretrizes da [RFC 5280](#).

Esta seção fornece exemplos para criar uma cadeia de autoridade de certificação intermediária usando duas abordagens diferentes: OpenSSL para geração de certificados locais e AWS Private Certificate Authority (PCA) para serviços gerenciados de certificados. Escolha a abordagem que melhor se adequa ao seu ambiente e aos requisitos de segurança.

Note

Os exemplos a seguir são casos de uso gerais e ambos são simplificados, usando a configuração mais básica. Para ambientes de produção, revise as opções adicionais de configuração e os requisitos de segurança específicos para seu caso de uso.

OpenSSL

Crie um arquivo de configuração do OpenSSL com extensões v3 comuns para CA:

```
$ cat > ca-extensions.conf <<EOF
[req]
distinguished_name = req_distinguished_name
[req_distinguished_name]
C = Country Name (2 letter code)
ST = State or Province Name (full name)
L = Locality Name (eg, city)
O = Organization Name (eg, company)
OU = Organizational Unit Name (eg, section)
CN = Common Name (e.g. server FQDN or YOUR name)
[v3_ca]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical,CA:true
keyUsage = critical, keyCertSign, cRLSign, digitalSignature
EOF
```

Gere uma CA raiz autoassinada usando o OpenSSL:

```
$ openssl req -new -x509 -days 3652 -key customerRootCA.key -out customerRootCA.crt
-extentions v3_ca -config ca-extensions.conf
Enter pass phrase for customerRootCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
```

For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:

Gere uma chave CA intermediária:

```
$ openssl genrsa -aes256 -out intermediateCA.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for intermediateCA.key:
Verifying - Enter pass phrase for intermediateCA.key:
```

Crie a solicitação intermediária de assinatura de certificado CA (CSR):

```
$ openssl req -new -key intermediateCA.key -out intermediateCA.csr
Enter pass phrase for intermediateCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Usando a CA raiz autoassinada, crie o certificado CA intermediário:

```
$ openssl x509 -req -in intermediateCA.csr \
  -CA customerRootCA.crt \
  -CAkey customerRootCA.key \
  -CAcreateserial \
  -days 3652 \
  -extensions v3_ca \
  -extfile ca-extensions.conf \
  -out intermediateCA.crt
```

```
Certificate request self-signature ok
subject=C= , ST= , L= , O= , OU=
```

Combine os certificados em um arquivo em cadeia:

```
$ cat intermediateCA.crt customerRootCA.crt > chainCA.crt

-----BEGIN CERTIFICATE-----
[Intermediate CA]
-----END CERTIFICATE-----
...
...
-----BEGIN CERTIFICATE-----
[Root CA]
-----END CERTIFICATE-----
```

Assine o CSR do cluster com sua CA intermediária:

```
$ openssl x509 -req -days 3652 -in <cluster ID>_ClusterCsr.csr \
  -CA intermediateCA.crt \
  -CAkey intermediateCA.key \
  -CAcreateserial \
  -out <cluster ID>_CustomerHsmCertificate.crt
```

```
Signature ok
subject=/C=US/ST=CA/O=Cavium/OU=N3FIPS/L=SanJose/CN=HSM:<HSM
  identifier>:PARTN:<partition number>, for FIPS mode
Getting CA Private Key
Enter pass phrase for intermediateCA.key:
```

AWS PCA

Crie e ative uma CA raiz usando a Autoridade de Certificação Privada da AWS:

```
$ # 1. Create Root CA
aws acm-pca create-certificate-authority \
  --certificate-authority-configuration \
    "KeyAlgorithm=RSA_4096,
    SigningAlgorithm=SHA256WITHRSA,

    Subject={Country=US,Organization=MyOrg,OrganizationalUnit=IT,CommonName=RootCA}" \
  --certificate-authority-type ROOT

# Store the Root CA Authority ARN from the previous output
ROOT_CA_AUTHORITY_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-
authority/<ca-authority-id>"

# 2. Generate Root CA CSR
aws acm-pca get-certificate-authority-csr \
  --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
  --output text > customerRootCA.csr

# 3. Self-sign Root CA Certificate
aws acm-pca issue-certificate \
  --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
  --csr fileb://customerRootCA.csr \
  --signing-algorithm SHA256WITHRSA \
  --template-arn arn:aws:acm-pca:::template/RootCACertificate/V1 \
  --validity Value=3652,Type=DAYS

# Store the Root CA certificate ARN from the previous output
ROOT_CA_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-authority/<ca-
authority-id>/certificate/<cert-id>"

# 4. Retrieve the Root CA certificate
aws acm-pca get-certificate \
  --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
  --certificate-arn $ROOT_CA_ARN \
  --output text > customerRootCA.crt

# 5. Import the Root CA Certificate
aws acm-pca import-certificate-authority-certificate \
  --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
  --certificate fileb://customerRootCA.crt
```

Crie e ative uma CA subordinada (também conhecida como CA intermediária):

```
$ # 6. Create Subordinate CA
aws acm-pca create-certificate-authority \
  --certificate-authority-configuration \
    "KeyAlgorithm=RSA_4096,
    SigningAlgorithm=SHA256WITHRSA,

    Subject={Country=US,Organization=MyOrg,OrganizationalUnit=IT,CommonName=SubordinateCA}" \
  --certificate-authority-type SUBORDINATE

# Store the Subordinate CA Authority ARN from the previous output
SUB_CA_AUTHORITY_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-
authority/<sub-ca-authority-id>"

# 7. Generate Subordinate CA CSR
aws acm-pca get-certificate-authority-csr \
  --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
  --output text > intermediateCA.csr

# 8. Issue Subordinate CA Certificate using Root CA
aws acm-pca issue-certificate \
  --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
  --csr fileb://intermediateCA.csr \
  --signing-algorithm SHA256WITHRSA \
  --template-arn arn:aws:acm-pca:::template/SubordinateCACertificate_PathLen0/V1 \
  --validity Value=3651,Type=DAYS

# Store the Subordinate CA certificate ARN from the previous output
SUB_CA_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-authority/<sub-ca-
authority-id>"

# 9. Retrieve Subordinate CA Certificate
aws acm-pca get-certificate \
  --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
  --certificate-arn $SUB_CA_ARN \
  --query 'Certificate' \
  --output text > intermediateCA.crt

# 10. Import the Subordinate CA Certificate
```

```
aws acm-pca import-certificate-authority-certificate \
  --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
  --certificate fileb://intermediateCA.crt \
  --certificate-chain fileb://customerRootCA.crt
```

Combine os certificados em um arquivo em cadeia:

```
$ cat intermediateCA.crt customerRootCA.crt > chainCA.crt

-----BEGIN CERTIFICATE-----
[Intermediate CA]
-----END CERTIFICATE-----
...
...
-----BEGIN CERTIFICATE-----
[Root CA]
-----END CERTIFICATE-----
```

Assine o CSR do cluster usando o AWS PCA:

```
$ aws acm-pca issue-certificate \
  --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
  --csr fileb://<cluster ID>_ClusterCsr.csr \
  --signing-algorithm SHA256WITHRSA \
  --template-arn arn:aws:acm-pca::template/EndEntityCertificate/V1 \
  --validity Value=3650,Type=DAYS

# Store your cluster's cert ARN from the previous output
CLUSTER_CERT_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-
authority/<cluster-cert-arn>"
```

Baixe o certificado de cluster assinado:

```
$ aws acm-pca get-certificate \
  --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
  --certificate-arn $CLUSTER_CERT_ARN \
  --output text --query Certificate > <cluster ID>_CustomerHsmCertificate.crt
```


Este comando cria um arquivo chamado `<cluster ID>_CustomerHsmCertificate.crt`. Use este arquivo como o certificado assinado ao inicializar o cluster.

Verifique o certificado assinado em relação à cadeia de certificados (opcional):

```
$ openssl verify -purpose sslserver -CAfile chainCA.crt <cluster ID>_CustomerHsmCertificate.crt
<cluster ID>_CustomerHsmCertificate.crt: OK
```

Depois de produzir o certificado HSM assinado com sua CA intermediária, acesse [Etapa 4: Inicializar o cluster](#).

Etapa 4: Inicializar o cluster

Use o certificado assinado do HSM e o certificado de assinatura para inicializar o cluster. Você pode usar o [AWS CloudHSM console AWS CLI](#), o ou a AWS CloudHSM API.

Console

Para inicializar um cluster (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione o botão de seleção ao lado do ID do cluster com o HSM que deseja verificar.
3. Selecione Ações. No menu suspenso, escolha Iniciar.
4. Se você não concluiu a [etapa anterior](#) para criar um HSM, escolha uma Zona de Disponibilidade (AZ) para o HSM que você está criando. Em seguida, selecione Criar.
5. Na página Download certificate signing request (Solicitação de assinatura de certificado de download), escolha Next (Próximo). Se a opção Next (Próximo) não estiver disponível, selecione primeiro um dos links de certificado ou de CSR. Em seguida, escolha Next (Próximo).
6. Na página de Download certificate signing request [Solicitação assinatura de certificado (CSR)], selecione Next (Próximo).
7. Na página Upload the certificates (Carregar os certificados), faça o seguinte:
 - a. Ao lado de Cluster certificate (Certificado de cluster) selecione Upload file (Carregar arquivo). Em seguida, localize e selecione o certificado do HSM assinado anteriormente. Se tiver executado as etapas na seção anterior, selecione o arquivo denominado `<cluster ID>_CustomerHsmCertificate.crt`.

- b. Ao lado de Issuing certificate (Certificado de emissão) selecione Upload file (Carregar arquivo). Em seguida, selecione seu certificado de assinatura com base na abordagem escolhida:
 - Se você escolheu a Opção A (certificado autoassinado único): selecione o arquivo chamado `<customerRootCA>.crt`
 - Se você escolheu a Opção B (cadeia de certificados): Selecione o arquivo chamado `<chainCA>.crt`
- c. Selecione Upload and initialize (Carregar e inicializar).

AWS CLI

Para inicializar um cluster ([AWS CLI](#))

- Em um prompt de comando, execute o comando [initialize-cluster](#). Forneça o seguinte:
 - O ID do cluster que foi criado anteriormente.
 - O certificado de HSM que foi assinado anteriormente. Se tiver executado as etapas na seção anterior, ele foi salvo em um arquivo denominado `<cluster ID>_CustomerHsmCertificate.crt`.
 - Seu certificado de assinatura com base na abordagem que você escolheu:
 - Se você escolheu a Opção A (certificado autoassinado único): use o arquivo chamado `<customerRootCA>.crt`
 - Se você escolheu a Opção B (cadeia de certificados): Use o arquivo chamado `<chainCA>.crt`

```
$ aws cloudhsmv2 initialize-cluster --cluster-id <cluster ID> \
                                     --signed-cert file://<cluster
                                     ID>_CustomerHsmCertificate.crt \
                                     --trust-anchor file://<customerRootCA.crt OR
                                     chainCA.crt>
{
  "State": "INITIALIZE_IN_PROGRESS",
  "StateMessage": "Cluster is initializing. State will change to INITIALIZED
upon completion."
}
```

AWS CloudHSM API

Para inicializar um cluster (AWS CloudHSM API)

- Envie uma solicitação [InitializeCluster](#) com o seguinte:
 - O ID do cluster que foi criado anteriormente.
 - O certificado de HSM que foi assinado anteriormente. Se tiver executado as etapas na seção anterior, ele foi salvo em um arquivo denominado *<cluster ID>_CustomerHsmCertificate.crt*.
 - Seu certificado de assinatura com base na abordagem que você escolheu:
 - Se você escolheu a Opção A (certificado autoassinado único): use o arquivo chamado *<customerRootCA>.crt*
 - Se você escolheu a Opção B (cadeia de certificados): Use o arquivo chamado *<chainCA>.crt*

Instalar e configurar a CLI do CloudHSM

Para interagir com o HSM em seu AWS CloudHSM cluster, você precisa da CLI do CloudHSM.

Conecte-se à sua instância cliente e execute os comandos a seguir para baixar e instalar as ferramentas da linha de AWS CloudHSM comando. Para obter mais informações, consulte [Inicie uma instância cliente do Amazon EC2 para interagir com AWS CloudHSM](#).

Amazon Linux 2023

Amazon Linux 2023 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

Amazon Linux 2023 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Amazon Linux 2 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.x86_64.rpm
```

Amazon Linux 2 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

RHEL 10 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-cli-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.x86_64.rpm
```

RHEL 10 sobre ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-cli-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

RHEL 9 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.x86_64.rpm
```

RHEL 9 sobre ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

RHEL 8 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.x86_64.rpm
```

RHEL 8 sobre ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.aarch64.rpm
```

Ubuntu 24.04 LTS

Ubuntu 24.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-cli_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_amd64.deb
```

Ubuntu 24.04 LTS na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-  
cli_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Ubuntu 22.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-  
cli_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_amd64.deb
```

Ubuntu 22.04 LTS na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-  
cli_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_arm64.deb
```

Windows Server 2022

Para o Windows Server 2022 na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/  
AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msixexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /  
quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2019

Para o Windows Server 2019 na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/  
AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msixec.exe -ArgumentList '/i C:\AWScloudHSMCLI-latest.msi /  
quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2016

Para o Windows Server 2016 na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/  
AWScloudHSMCLI-latest.msi -Outfile C:\AWScloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msixec.exe -ArgumentList '/i C:\AWScloudHSMCLI-latest.msi /  
quiet /norestart /log C:\client-install.txt' -Wait
```

Use o comando a seguir para configurar a CloudHSM CLI.

Para inicializar uma EC2 instância Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the  
HSMs>
```

Para inicializar uma EC2 instância do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI  
IPv4 / IPv6 addresses of the HSMs>
```

Ative o cluster em AWS CloudHSM

Quando você ativa um AWS CloudHSM cluster, o estado do cluster muda de inicializado para ativo. Em seguida, é possível [gerenciar os usuários do HSM](#) e [usar o HSM](#).

 Important

Antes de ativar o cluster, você deve primeiro copiar o certificado de emissão para o local padrão da plataforma em cada EC2 instância que se conecta ao cluster (você cria o certificado de emissão ao inicializar o cluster). Use o arquivo de certificado apropriado com base na abordagem escolhida durante a inicialização do cluster:

- Se você escolheu a Opção A (certificado autoassinado único): Copiar `customerRootCA.crt`
- Se você escolheu a Opção B (cadeia de certificados): Copiar `chainCA.crt`

Localização do Linux:

```
/opt/cloudhsm/etc/<customerRootCA.crt OR chainCA.crt>
```

Localização do Windows:

```
C:\ProgramData\Amazon\CloudHSM\<customerRootCA.crt OR chainCA.crt>
```

Depois de copiar o arquivo do certificado, edite o `/opt/cloudhsm/etc/cloudhsm-cli.cfg` arquivo para garantir que o nome do arquivo do certificado corresponda ao nome do certificado CA que você copiou.

Depois de colocar o certificado de emissão, instale a CLI do CloudHSM e execute o comando [cluster activate](#) em seu primeiro HSM. Você notará que a conta de administrador no primeiro HSM em seu cluster tem a função de administrador [não ativada](#). Essa é uma função temporária que só existe antes da ativação do cluster. Quando você ativa seu cluster, a função de administrador não ativado muda para administrador.

Para ativar um cluster

1. Conecte-se à instância do cliente que você iniciou anteriormente. Para obter mais informações, consulte [Inicie uma instância cliente do Amazon EC2 para interagir com AWS CloudHSM](#). Você pode iniciar uma instância do Linux ou um Windows Server.
2. Execute a CLI do CloudHSM no modo interativo.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

3. (Opcional) Use o comando `user list` para exibir os usuários existentes.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "unactivated-admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

4. Use o comando `cluster activate` para definir a senha inicial do administrador.

```
aws-cloudhsm > cluster activate
Enter
password:<NewPassword>
Confirm password:<NewPassword>
{
  "error_code": 0,
```

```
"data": "Cluster activation successful"
}
```

Recomendamos anotar a nova senha em uma planilha de senhas. Não perca a planilha. Recomendamos que você imprima uma cópia da planilha de senhas, use-a para registrar as senhas críticas do HSM e armazene-a em um local seguro. Também recomendamos armazenar uma cópia dessa planilha em um local externo seguro.

5. (Opcional) Use o comando `user list` para verificar se o tipo de usuário foi alterado para [administrador/responsável pela criptografia \(CO\)](#).

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

6. Use o comando `quit` para parar a ferramenta CLI do CloudHSM.

```
aws-cloudhsm > quit
```

Para obter mais informações sobre como trabalhar com a CLI do CloudHSM ou a CMU, consulte [Entendendo os usuários do HSM](#) e [Compreendendo o gerenciamento de usuários do HSM com o CMU](#).

Configurar o TLS mútuo entre o cliente e AWS CloudHSM (recomendado)

Os tópicos a seguir descrevem as etapas que você deve concluir para habilitar o Transport Layer Security (mTLS) mútuo entre o cliente e AWS CloudHSM.

Considerações

- Atualmente, esse recurso está disponível exclusivamente no hsm2m.medium. Para obter mais informações sobre o tipo de HSM, consulte [AWS CloudHSM modos de cluster](#).
- O mTLS não é compatível com armazenamentos de AWS CloudHSM chaves usados com AWS Key Management Service.

Tópicos

- [Etapa 1. Criar e registrar uma âncora de confiança no HSM](#)
- [Etapa 2. Habilite o mTLS para AWS CloudHSM](#)
- [Etapa 3. Defina a aplicação do mTLS para AWS CloudHSM](#)

Etapa 1. Criar e registrar uma âncora de confiança no HSM

Uma âncora de confiança deve ser criada e registrada no HSM antes de ativar o mTLS. Este é um processo em duas etapas:

Tópicos

- [Criar uma chave privada e um certificado raiz autoassinado](#)
- [Registrar a âncora de confiança no HSM](#)

Criar uma chave privada e um certificado raiz autoassinado

Note

Para um cluster de produção, a chave deve ser criada com segurança usando uma fonte confiável de aleatoriedade. Recomendamos que você use um HSM fora do local e off-line ou o equivalente. Armazene a chave com segurança.

Durante a fase de desenvolvimento e teste, você pode usar qualquer ferramenta conveniente (como o OpenSSL) para criar a chave e autoassinar um certificado raiz. Você precisará da chave e do certificado raiz para assinar o certificado do cliente no comando [enable mTLS for AWS CloudHSM](#).

Os exemplos a seguir mostram como criar uma chave privada e um certificado raiz autoassinado com o [OpenSSL](#).

Example Crie uma chave privada com o OpenSSL

Use o comando a seguir para criar uma chave de RSA de 4096 bits criptografada com o algoritmo AES-256. Para usar esse exemplo, `<mtls_ca_root_1.key>` substitua pelo nome do arquivo em que você deseja armazenar a chave.

```
$ openssl genrsa -out <mtls_ca_root_1.key> -aes256 4096
Generating RSA private key, 4096 bit long modulus
.....+++
.+++
e is 65537 (0x10001)
Enter pass phrase for mtls_ca_root_1.key:
Verifying - Enter pass phrase for mtls_ca_root_1.key:
```

Example– Criar um certificado raiz autoassinado com o OpenSSL

Use o comando a seguir para criar um certificado raiz autoassinado chamado `mtls_ca_root_1.crt` da chave privada que você acabou de criar. O certificado é válido por 25 anos (9130 dias). Leia as instruções na tela e siga os avisos.

```
$ openssl req -new -x509 -days 9130 -key mtls_ca_root_1.key -out mtls_ca_root_1.crt
Enter pass phrase for mtls_ca_root_1.key:
You are about to be asked to enter information that will be incorporated
```

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:

State or Province Name (full name) [Some-State]:

Locality Name (eg, city) []:

Organization Name (eg, company) [Internet Widgits Pty Ltd]:

Organizational Unit Name (eg, section) []:

Common Name (e.g. server FQDN or YOUR name) []:

Email Address []:

Registrar a âncora de confiança no HSM

Depois de criar um certificado raiz autoassinado, o administrador deve registrá-lo como âncora confiável no AWS CloudHSM cluster.

Para registrar uma âncora de confiança no HSM

1. Use o comando a seguir para iniciar CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CloudHSM CLI, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Use o comando [Registrar uma âncora de confiança com a CLI do CloudHSM](#) para registrar a âncora de confiança. Para obter mais informações, consulte o exemplo a seguir ou use o comando `help cluster mtls register-trust-anchor`.

Example— Registre uma âncora de confiança com o cluster AWS CloudHSM

O exemplo a seguir mostra como usar o comando `cluster mtls register-trust-anchor` na CLI do CloudHSM para registrar uma âncora de confiança no HSM. Para usar esse comando, o CO deve estar conectado no HSM. Substitua estes valores pelos seus próprios:

```
aws-cloudhsm > cluster mtls register-trust-anchor --path </path/mtls_ca_root_1.crt>
{
  "error_code": 0,
  "data": {
    "trust_anchor": {
      "certificate-reference": "0x01",
      "certificate": "<PEM Encoded Certificate>",
      "cluster-coverage": "full"
    }
  }
}
```

 Note

AWS CloudHSM suporta o registro de certificados intermediários como âncora confiável. Nesses casos, todo o arquivo da cadeia de certificados codificado pelo PEM precisa ser registrado no HSM, com os certificados em ordem hierárquica. AWS CloudHSM suporta uma cadeia de certificados de 6980 bytes.

Depois de registrar com êxito a âncora de confiança, é possível executar o comando `cluster mtls list-trust-anchors` para verificar as âncoras de confiança registradas atuais, conforme mostrado abaixo:

```
aws-cloudhsm > cluster mtls list-trust-anchors
{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
```

```

    "certificate": "<PEM Encoded Certificate>",
    "cluster-coverage": "full"
  }
]
}
}

```

Note

No máximo 2 (duas) âncoras de confiança podem ser registradas em instâncias hsm2m.medium.

Etapa 2. Habilite o mTLS para AWS CloudHSM

Para habilitar o mTLS para AWS CloudHSM, você precisa criar uma chave privada e um certificado de cliente assinado pelo certificado raiz que geramos em [Criar e registrar uma âncora confiável no HSM e, em seguida](#), usar qualquer ferramenta de configuração do Client SDK 5 para configurar o caminho da chave privada e o caminho da cadeia de certificados do cliente.

Tópicos

- [Criar uma chave privada e uma cadeia de certificados de clientes](#)
- [Configurar mTLS para Client SDK 5](#)

Criar uma chave privada e uma cadeia de certificados de clientes

Example Crie uma chave privada com o OpenSSL

Use o comando a seguir para criar uma chave RSA de 4096 bits. Para usar esse exemplo, *<ssl-client.key>* substitua pelo nome do arquivo em que você deseja armazenar a chave.

```

$ openssl genrsa -out <ssl-client.key> 4096
Generating RSA private key, 4096 bit long modulus
.....+++
.+++
e is 65537 (0x10001)

```

Example– Gere uma Certificate Signing Request (CSR – Solicitação de assinatura de certificado).

Use o comando a seguir para gerar uma Certificate Signing Request (CSR – Solicitação de assinatura de certificado) com a chave privada que você acabou de criar. Leia as instruções na tela e siga os avisos.

```
$ openssl req -new -key <ssl-client.key> -out <ssl-client.csr>
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Example– Assinar a CSR com o certificado raiz

Use o comando a seguir para assinar a CSR com o certificado raiz que criamos e registramos em [Criar e registrar uma âncora de confiança no HSM](#) e criar um certificado de cliente chamado `ssl-client.crt`. O certificado é válido por 5 anos (1826 dias).

```
$ openssl x509 -req -days 1826 -in <ssl-client.csr> -CA <mtls_ca_root_1.crt> -
CAkey <mtls_ca_root_1.key> -CAcreateserial -out <ssl-client.crt>
```

Example– Criar uma cadeia de certificado de cliente

Use o comando a seguir para combinar o certificado do cliente e o certificado raiz que criamos e registramos em [Criar e registrar uma âncora de confiança no HSM](#) e criar uma cadeia de certificados do cliente chamada `ssl-client.pem`, que será usada para configurar na próxima etapa.

```
$ cat <ssl-client.crt> <mtls_ca_root_1.crt> > <ssl-client.pem>
```


Note

Se você registrou certificados intermediários em [Criar e registrar uma âncora de confiança no HSM](#) como âncora de confiança, combine o certificado do cliente com toda a cadeia de certificados para criar uma cadeia de certificados do cliente.

Configurar mTLS para Client SDK 5

Use qualquer ferramenta de configuração do Client SDK 5 para habilitar o TLS mútuo fornecendo o caminho correto da chave do cliente e o caminho da cadeia de certificados do cliente. Para obter mais informações sobre a ferramenta de configuração para o Client SDK 5, consulte [???](#).

PKCS #11 library

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
$ sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
    --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
    --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
    --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
    --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

OpenSSL Dynamic Engine

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
    --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
    --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Key Storage Provider (KSP)

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" `
    --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
```

```
--client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-  
client.key>
```

JCE provider

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>  
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-jce \  
--client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \  
--client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>  
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" \  
--client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-  
client.pem> \  
--client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-  
client.key>
```

CloudHSM CLI

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-cli \
    --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
    --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
    --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
    --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

Etapa 3. Defina a aplicação do mTLS para AWS CloudHSM

Depois de configurar com qualquer uma das ferramentas de configuração do Client SDK 5, a conexão entre o cliente e o cliente AWS CloudHSM será TLS mútuo no cluster. No entanto, remover o caminho da chave privada e o caminho da cadeia de certificados do cliente do arquivo

de configuração transformará a conexão em TLS normal novamente. É possível usar a CLI do CloudHSM para definir a imposição de mTLS no cluster concluindo as seguintes etapas:

1. Use o comando a seguir para iniciar CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CloudHSM CLI, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

Note

1. Verifique se você configurou a CLI do CloudHSM e inicie-a em uma conexão mTLS.
2. É preciso estar logado como usuário administrador padrão com o nome de usuário administrador antes de definir a imposição do mTLS.

3. Use o comando [Definir o nível de imposição do mTLS com a CloudHSM CLI](#) para definir a imposição. Para obter mais informações, consulte o exemplo a seguir ou use o comando `help cluster mtls set-enforcement`.

Example— Defina a aplicação do mTLS com cluster AWS CloudHSM

O exemplo a seguir mostra como usar o comando `cluster mtls set-enforcement` na CLI do CloudHSM a fim de definir a imposição de mTLS com o HSM. Para usar esse comando, o administrador com nome de usuário admin deve estar conectado no HSM.

```
aws-cloudhsm > cluster mtls set-enforcement --level cluster
{
  "error_code": 0,
  "data": {
    "message": "Mtls enforcement level set to Cluster successfully"
  }
}
```

 Warning

Depois de impor o uso de mTLS no cluster, todas as conexões não MTLS existentes serão descartadas e a conexão com o cluster será possível apenas com certificados mTLS.

Crie e use chaves em AWS CloudHSM

[Antes de criar e usar chaves em seu novo cluster, crie um usuário do módulo de segurança de hardware \(HSM\) com a CLI do AWS CloudHSM. Para obter mais informações, consulte Entendendo as tarefas de gerenciamento de usuários do HSM, Introdução à interface de linha de comando \(CLI\) do AWS CloudHSM e Como gerenciar usuários do HSM.](#)

 Note

Se estiver usando o SDK do cliente 3, use o [CloudHSM Management Utility \(CMU\)](#) em vez do CloudHSM CLI.

Depois de criar usuários do HSM, você pode entrar no HSM e gerenciar as chaves usando qualquer uma destas opções:

- Use o [utilitário de gerenciamento de chaves, uma ferramenta de linha de comando](#)
- Crie um aplicativo C usando a biblioteca [PKCS #11](#)
- Crie um aplicativo Java usando o [provedor JCE](#)
- Use o [OpenSSL Dynamic Engine diretamente da linha de comando](#)
- Use o OpenSSL Dynamic Engine para descarregamento de TLS com [servidores web NGINX e Apache](#)

- Use o Provedor de Armazenamento de Chaves (KSP) para AWS CloudHSM com a [Autoridade de Certificação \(CA\) do Microsoft Windows Server](#)
- Use o Key Storage Provider (KSP) para usar a AWS CloudHSM ferramenta [Microsoft Sign](#)
- Use os provedores de CNG e KSP para descarga de TLS com o servidor Web do [Internet Information Server \(IIS\)](#)

Melhores práticas para AWS CloudHSM

Execute as melhores práticas deste tópico para usar o AWS CloudHSM com eficiência.

Conteúdo

- [AWS CloudHSM melhores práticas de gerenciamento de clusters](#)
- [AWS CloudHSM melhores práticas de gerenciamento de usuários](#)
- [AWS CloudHSM melhores práticas de gerenciamento de chaves](#)
- [AWS CloudHSM melhores práticas de integração de aplicativos](#)
- [AWS CloudHSM melhores práticas de monitoramento](#)

AWS CloudHSM melhores práticas de gerenciamento de clusters

Siga as melhores práticas desta seção ao criar, acessar e gerenciar seu AWS CloudHSM cluster.

Escale seu cluster para lidar com picos de tráfego

Vários fatores podem influenciar a throughput máxima que seu cluster pode suportar, incluindo tamanho da instância do cliente, tamanho do cluster, topografia da rede e as operações criptográficas necessárias para seu caso de uso.

Como ponto de partida, consulte o tópico [AWS CloudHSM informações de desempenho](#) para obter estimativas de desempenho sobre tamanhos e configurações comuns de clusters. Recomendamos que você teste a carga do seu cluster com o pico de carga previsto para determinar se sua arquitetura atual é resiliente e está na escala certa.

Arquitete seu cluster para alta disponibilidade

Adicione redundância à contabilização da manutenção: AWS pode substituir seu HSM para manutenção programada ou se ele detectar um problema. Como regra geral, o tamanho do seu cluster deve ter pelo menos +1 de redundância. Por exemplo, se você precisar de dois HSMs para que seu serviço opere nos horários de pico, o tamanho ideal do cluster será três. Se você seguir as melhores práticas relacionadas à disponibilidade, essas substituições de HSM não devem afetar seu serviço. No entanto, as operações em andamento no HSM substituído podem falhar e devem ser repetidas.

Espalhe você HSMs por várias zonas de disponibilidade: considere como seu serviço poderá operar durante uma interrupção na zona de disponibilidade. AWS recomenda que você as HSMs distribua pelo maior número possível de zonas de disponibilidade. Para um cluster com três HSMs, você deve se HSMs espalhar por três zonas de disponibilidade. Dependendo do seu sistema, você pode precisar de redundância adicional.

Tenha pelo menos três HSMs para garantir a durabilidade das chaves recém-geradas

Para aplicativos que exigem durabilidade de chaves recém-geradas, recomendamos ter pelo menos três HSMs espalhadas por diferentes zonas de disponibilidade em uma região.

Acesso seguro ao cluster

Use sub-redes privadas para limitar o acesso à sua instância: execute suas instâncias HSMs e as do cliente nas sub-redes privadas da sua VPC. Isso limita o acesso ao seu HSMs do mundo exterior.

Use endpoints VPC para acessar APIs: o plano de AWS CloudHSM dados foi projetado para operar sem a necessidade de acesso à Internet ou à AWS. APIs Se sua instância cliente exigir acesso à AWS CloudHSM API, você poderá usar VPC endpoints para acessar a API sem precisar de acesso à Internet em sua instância cliente. Consulte [AWS CloudHSM e VPC endpoints](#) para obter mais informações.

Reduza os custos escalando de acordo com suas necessidades

Não há custos iniciais de uso do AWS CloudHSM. Você paga uma taxa horária por cada HSM lançado até encerrar o HSM. Se o seu serviço não exigir o uso contínuo do AWS CloudHSM, você pode reduzir os custos reduzindo (excluindo) seus HSMs para zero quando eles não forem necessários. Quando HSMs necessário novamente, você pode restaurá-lo a HSMs partir de um backup. Se, por exemplo, você tiver uma carga de trabalho exigindo que você assine o código uma vez por mês, especificamente no último dia do mês, você pode escalar seu cluster antes, reduzi-lo excluindo o seu HSMs após a conclusão do trabalho e, em seguida, restaurar seu cluster para realizar operações de assinatura novamente no final do próximo mês.

AWS CloudHSM faz automaticamente backups periódicos do HSMs no cluster. Ao adicionar um novo HSM em uma data posterior, AWS CloudHSM restaurará o backup mais recente no novo HSM para que você possa retomar o uso do mesmo local em que o deixou. Para calcular seus custos de AWS CloudHSM arquitetura, consulte [AWS CloudHSM Preços](#).

Recursos relacionados:

- [Visão geral dos backups](#)
- [Backup retention policy \(Política de retenção de backup\)](#)
- [Copiar backups de AWS CloudHSM cluster entre AWS regiões](#)

AWS CloudHSM melhores práticas de gerenciamento de usuários

Siga as melhores práticas nesta seção para gerenciar com eficiência os usuários em seu AWS CloudHSM cluster. Os usuários do HSM são diferentes dos usuários do IAM. Usuários e entidades do IAM que têm uma política baseada em identidade com as permissões apropriadas podem criar interagindo com recursos HSMs por meio da API da AWS. Depois que o HSM for criado, você deverá usar as credenciais de usuário do HSM para autenticar as operações no HSM. Para obter um guia detalhado dos usuários do HSM, consulte [Usuários do HSM em AWS CloudHSM](#).

Proteja as credenciais de seus usuários do HSM

É fundamental manter as credenciais de seus usuários do HSM protegidas com segurança, pois eles são as entidades que podem acessar e realizar operações criptográficas e de gerenciamento no seu HSM. AWS CloudHSM não tem acesso às suas credenciais de usuário do HSM e não poderá ajudá-lo se você perder o acesso a elas.

Tenha pelo menos dois administradores para evitar o bloqueio

Para evitar ficar bloqueado em seu cluster, recomendamos que você tenha pelo menos dois administradores, caso uma senha de administrador seja perdida. Caso isso aconteça, você pode usar o outro administrador para redefinir a senha.

Note

Administradores no SDK do Cliente 5 são sinônimos de agentes criptográficos (COs) no SDK do Cliente 3.

Habilite o quórum para todas as operações de gerenciamento de usuários

O quórum permite que você defina um número mínimo de administradores que devem aprovar uma operação de gerenciamento de usuários antes que essa operação possa ocorrer. Devido

ao privilégio que os administradores têm, recomendamos que você habilite o quórum para todas as operações de gerenciamento de usuários. Isso pode limitar o potencial de impacto se uma de suas senhas de administrador for comprometida. Para obter mais informações, consulte [Gerenciar quórum](#).

Crie vários usuários de criptografia, cada um com permissões limitadas

Ao separar as responsabilidades dos usuários de criptografia, nenhum usuário tem controle total sobre todo o sistema. Por esse motivo, recomendamos que você crie vários usuários de criptografias e limite as permissões de cada um. Normalmente, isso é feito atribuindo a diferentes usuários de criptografia responsabilidades e ações distintas que eles realizam (por exemplo, ter um usuário de criptografia responsável por gerar e compartilhar chaves com outros usuários de criptografia que, em seguida, as utilizam em seu aplicativo).

Recursos relacionados:

- [Compartilhar uma chave usando a CloudHSM CLI](#)
- [Cancelar o compartilhamento de uma chave usando a CloudHSM CLI](#)

AWS CloudHSM melhores práticas de gerenciamento de chaves

Siga as práticas mais recomendadas desta seção ao gerenciar as chaves em AWS CloudHSM.

Escolha o tipo de chave certo

Ao usar uma chave de sessão, suas transações por segundo (TPS) serão limitadas a um HSM onde a chave existe. Extras HSMs em seu cluster não aumentarão a taxa de transferência das solicitações dessa chave. Se você usar uma chave de token para o mesmo aplicativo, suas solicitações terão a carga balanceada entre todas as disponíveis HSMs em seu cluster. Para obter mais informações, consulte [Configurações de sincronização e durabilidade de teclas em AWS CloudHSM](#).

Gerencie os principais limites de armazenamento

HSMs têm limites no número máximo de tokens e chaves de sessão que podem ser armazenados em um HSM ao mesmo tempo. Para obter informações sobre limites de armazenamento, consulte [AWS CloudHSM cotas](#). Se seu aplicativo exigir mais do que o limite, você poderá usar uma ou mais das seguintes estratégias para gerenciar as chaves com eficiência:

Use um encapsulamento confiável para armazenar suas chaves em um armazenamento de dados externo: usando o agrupamento de chaves confiável, você pode superar o limite de armazenamento de chaves armazenando todas as chaves em um armazenamento de dados externo. Quando precisar usar essa chave, você pode desencapsular a chave no HSM como uma chave de sessão, usar a chave para a operação necessária e, em seguida, descartar a chave de sessão. Os dados-chave originais permanecem armazenados com segurança em seu armazenamento de dados para uso sempre que você precisar. Usar chaves confiáveis para fazer isso maximiza sua proteção.

Distribua chaves entre clusters: outra estratégia para superar o limite de armazenamento de chaves é armazenar suas chaves em vários clusters. Nessa abordagem, você mantém um mapeamento das chaves armazenadas em cada cluster. Use esse mapeamento para rotear as solicitações do seu cliente para o cluster com a chave necessária. Para obter informações sobre como se conectar a vários clusters do mesmo aplicativo cliente, consulte os tópicos a seguir:

- [Conectando-se a vários AWS CloudHSM clusters com o provedor JCE](#)
- [Configuração de vários slots com a biblioteca PKCS #11 para AWS CloudHSM](#)

Gerenciando e protegendo o encapsulamento de chaves

As chaves podem ser marcadas como extraíveis ou não extraíveis por meio do atributo `EXTRACTABLE`. Por padrão, as chaves HSM são marcadas como extraíveis.

Chaves extraíveis são chaves que podem ser exportadas do HSM por meio do encapsulamento de chaves. As chaves encapsuladas são criptografadas e devem ser desencapsuladas usando a mesma chave de encapsulamento antes de poderem ser usadas. Chaves não extraíveis não podem ser exportadas do HSM em nenhuma circunstância. Não há como tornar extraível uma chave não extraível. Por esse motivo, é importante considerar se você exige que suas chaves sejam extraíveis ou não e definir o atributo de chave correspondente adequadamente.

Se você precisar de agrupamento de chaves em seu aplicativo, deverá utilizar o agrupamento de chaves confiável para limitar a capacidade dos usuários do HSM somente a wrap/unwrap chaves que tenham sido explicitamente marcadas como confiáveis por um administrador. Para obter mais informações, consulte tópicos sobre encapsulamento de chaves confiáveis em [Entrada de chaves AWS CloudHSM](#).

Recursos relacionados

- [Funções de agrupamento e desagrupamento](#)

- [Funções de criptografia para JCE](#)
- [Atributos de chave Java compatíveis com o AWS CloudHSM Client SDK 5](#)
- [Atributos de chave da CloudHSM CLI](#)

AWS CloudHSM melhores práticas de integração de aplicativos

Siga as melhores práticas nesta seção para otimizar a integração do aplicativo com o AWS CloudHSM cluster.

Faça o bootstrap do seu Client SDK

Antes que seu SDK do cliente possa se conectar ao seu cluster, deve ser feito o bootstrap. Ao fazer bootstrap dos endereços IP para seu cluster, recomendamos usar o parâmetro `--cluster-id` sempre que possível. Esse método preenche sua configuração com todos os endereços IP do HSM em seu cluster sem precisar acompanhar cada endereço individual. Isso adiciona mais resiliência à inicialização do seu aplicativo no caso de um HSM estar em manutenção ou durante uma interrupção na zona de disponibilidade. Consulte mais detalhes em [Bootstrap o Client SDK](#).

Autentique-se para realizar operações

Em AWS CloudHSM, você deve se autenticar em seu cluster antes de poder realizar a maioria das operações, como operações criptográficas.

Autenticar com a CloudHSM CLI: você pode se autenticar com a CloudHSM CLI usando seu [modo de comando único](#) ou o [modo interativo](#). Use o comando [Fazer login em um HSM usando a CLI do CloudHSM](#) para autenticar no modo interativo. Para autenticar no modo de comando único, você deve definir as variáveis ambientais `CLOUDHSM_ROLE` e `CLOUDHSM_PIN`. Para obter detalhes sobre como fazer isso, consulte [Modo de comando único](#). O AWS CloudHSM recomenda armazenar com segurança suas credenciais do HSM quando não estiverem sendo usadas pela sua aplicação.

Autenticar com PKCS #11: No PKCS #11, você faz login usando a API `C_Login` depois de abrir uma sessão usando `C_`. `OpenSession`. Você só precisa executar um `C_Login` por slot (cluster). Depois de fazer login com sucesso, você pode abrir sessões adicionais usando `C_` `OpenSession` sem a necessidade de realizar operações adicionais de login. Para obter exemplos de autenticação no PKCS #11, consulte [Exemplos de código para a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#).

Autenticar com o JCE: O provedor AWS CloudHSM JCE suporta login implícito e explícito. O método que você escolhe depende do seu caso de uso. Quando possível, recomendamos o uso do Login Implícito, pois o SDK processará automaticamente a autenticação se seu aplicativo for desconectado do cluster e precisar ser autenticado novamente. O uso do login implícito também permite que você forneça credenciais ao seu aplicativo ao usar uma integração que não permite que você tenha controle sobre o código do aplicativo. Para obter mais informações sobre os métodos de login, consulte [Etapa 2: fornecer credenciais ao provedor JCE](#).

Autenticar com OpenSSL: com o OpenSSL Dynamic Engine, você fornece credenciais por meio de variáveis de ambiente. O AWS CloudHSM recomenda armazenar com segurança suas credenciais do HSM quando não estiverem sendo usadas pela sua aplicação. Se possível, você deve configurar seu ambiente para recuperar e definir sistematicamente essas variáveis de ambiente sem entrada manual. Para obter detalhes sobre a autenticação com o OpenSSL, consulte [Instale o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5](#).

Autenticar com o KSP: é possível se autenticar com o Provedor de armazenamento de chaves (KSP) usando o gerenciador de credenciais do Windows ou variáveis de ambiente, consulte [Instale o provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#).

Gerencie com eficácia as chaves em seu aplicativo

Use os atributos-chave para controlar o que as chaves podem fazer: ao gerar uma chave, use os atributos-chave para definir um conjunto de permissões que permitirão ou negarão tipos específicos de operações para essa chave. Recomendamos que as chaves sejam geradas com a menor quantidade de atributos necessária para concluir a tarefa. Por exemplo, uma chave AES usada para criptografia também não deve ter permissão para extrair chaves do HSM. Para obter mais informações, consulte nossas páginas de atributos para o seguinte cliente SDKs:

- [Atributos de chaves do PKCS #11](#)
- [Atributos de chaves do JCE](#)

Quando possível, armazene em cache os objetos de chaves para minimizar a latência: as operações de localização de chaves consultarão cada HSM em seu cluster. Essa operação é cara e não se escala com a contagem de HSM em seu cluster.

- Com o PKCS #11, você encontra chaves usando a API do `C_FindObjects`.
- Com o JCE, você encontra chaves usando o `KeyStore`

Para um desempenho ideal, AWS recomenda que você utilize comandos de localização de chave (como [Pesquise AWS CloudHSM chaves por atributos usando KMU](#) e [Listar chaves para um usuário com a CloudHSM CLI](#)) somente uma vez durante a inicialização do aplicativo e armazene em cache o objeto chave retornado na memória do aplicativo. Se você precisar desse objeto chave posteriormente, deverá recuperar o objeto do seu cache em vez de consultar esse objeto para cada operação, o que adicionará uma sobrecarga significativa de desempenho.

Use multiencadeamento

AWS CloudHSM oferece suporte a aplicativos multiencadeados, mas há algumas coisas a serem lembradas com aplicativos multiencadeados.

Com o PKCS #11, você deve inicializar a biblioteca PKCS #11 (chamando `C_Initialize`) somente uma vez. Cada thread deve ter sua própria sessão (`C_OpenSession`) atribuída. Não é recomendável usar a mesma sessão em vários threads.

Com o JCE, o AWS CloudHSM provedor deve ser inicializado somente uma vez. Não compartilhe instâncias de objetos SPI entre threads. Por exemplo, Cipher, Signature, Digest, Mac KeyFactory ou KeyGenerator objetos só devem ser utilizados no contexto de seu próprio thread.

Lide com erros de controle de utilização

Você pode enfrentar erros de controle de utilização do HSM nas seguintes circunstâncias:

- Seu cluster não está adequadamente escalado para gerenciar picos de tráfego.
- Seu cluster não está dimensionado com redundância de +1 durante eventos de manutenção.
- Interrupções na zona de disponibilidade resultam em um número reduzido de itens disponíveis HSMs em seu cluster.

Consulte [Controle de utilização do HSM](#) para obter informações sobre a melhor forma de lidar com esse cenário.

Para garantir que seu cluster tenha o tamanho adequado e não seja limitado, AWS recomenda que você faça um teste de carga em seu ambiente com o pico de tráfego esperado.

Integre novas tentativas em operações de cluster

AWS pode substituir seu HSM por motivos operacionais ou de manutenção. Para tornar seu aplicativo resiliente a essas situações, AWS recomenda que você implemente a lógica de repetição

do lado do cliente em todas as operações que são roteadas para seu cluster. Espera-se que novas tentativas de operações com falha devido a substituições sejam bem-sucedidas.

Implementação de estratégias de recuperação de desastres

Em resposta a um evento, pode ser necessário afastar seu tráfego de um cluster ou região inteira. As seções a seguir descrevem várias estratégias para fazer isso.

Use o emparelhamento de VPC para acessar seu cluster de outra conta ou região: você pode utilizar o emparelhamento de VPC para acessar seu AWS CloudHSM cluster de outra conta ou região. Para ter mais informações, consulte [O que é emparelhamento de VPC?](#) no Amazon VPC Peering Guide. Depois de estabelecer suas conexões de peering e configurar seus grupos de segurança adequadamente, você poderá se comunicar com os endereços IP do HSM da mesma forma que faria normalmente.

Conecte-se a vários clusters da mesma aplicação: o provedor JCE, a biblioteca PKCS #11 e a CloudHSM CLI no Client SDK 5 oferecem suporte à conexão com vários clusters da mesma aplicação. Por exemplo, você pode ter dois clusters ativos, cada um em regiões diferentes, e seu aplicativo pode se conectar a ambos ao mesmo tempo e equilibrar a carga entre os dois como parte das operações normais. Se seu aplicativo não estiver usando o Client SDK 5 (o SDK mais recente), você não poderá se conectar a vários clusters do mesmo aplicativo. Como alternativa, você pode manter outro cluster em funcionamento e, no caso de uma interrupção regional, transferir seu tráfego para o outro cluster para minimizar o tempo de inatividade. Consulte as respectivas páginas para obter detalhes:

- [Configuração de vários slots com a biblioteca PKCS #11 para AWS CloudHSM](#)
- [Conectando-se a vários AWS CloudHSM clusters com o provedor JCE](#)
- [Conectar-se a vários clusters com a CLI do CloudHSM](#)

Restaurar um cluster a partir de um backup: você pode criar um novo cluster a partir do backup de um cluster existente. Para obter mais informações, consulte [Backups de cluster em AWS CloudHSM](#).

Escalone a implantação de seu aplicativo

Siga estratégias de implantação escalonadas, como implantações progressivas baseadas em ondas, implantações únicas e implantações contínuas para implantações e reinicializações de aplicativos clientes. Essa abordagem minimiza o impacto potencial das mudanças e, ao mesmo tempo, garante que haja capacidade suficiente para atender ao tráfego de produção durante a implantação.

AWS CloudHSM melhores práticas de monitoramento

Esta seção descreve vários mecanismos que você pode usar para monitorar seu cluster e aplicativo. Para obter detalhes adicionais sobre monitoramento, consulte [Monitoramento AWS CloudHSM](#).

Monitoramento de logs do cliente

Cada Client SDK grava registros que você pode monitorar. Para obter mais informações sobre registro em log, consulte [Trabalhando com registros AWS CloudHSM do SDK do cliente](#).

Em plataformas projetadas para serem efêmeras, como Amazon ECS e AWS Lambda, coletar registros de clientes de um arquivo pode ser difícil. Nessas situações, é uma prática recomendada configurar o registro do Client SDK para gravar logs no console. A maioria dos serviços coletará automaticamente essa saída e a publicará CloudWatch nos registros da Amazon para você manter e visualizar.

Se você estiver usando qualquer integração de terceiros além do SDK do AWS CloudHSM cliente, certifique-se de configurar esse pacote de software para registrar também sua saída no console. Caso contrário, a saída do SDK do AWS CloudHSM cliente pode ser capturada por esse pacote e gravada em seu próprio arquivo de log.

Consulte o [AWS CloudHSM Ferramenta de configuração do Client SDK 5](#) para obter informações sobre como configurar as opções de registro em seu aplicativo.

Monitoramento de logs de auditoria

AWS CloudHSM publica registros de auditoria na sua CloudWatch conta da Amazon. Os registros de auditoria vêm do HSM e rastreiam determinadas operações para fins de auditoria.

Você pode usar registros de auditoria para acompanhar todos os comandos de gerenciamento que são invocados no seu HSM. Por exemplo, você pode acionar um alarme ao perceber que uma operação de gerenciamento inesperada está sendo executada.

Consulte [Como funciona o registro em log de auditoria do HSM](#) para obter mais detalhes.

Monitorar AWS CloudTrail

AWS CloudHSM é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço em AWS CloudHSM. AWS CloudTrail captura

todas as chamadas de API AWS CloudHSM como eventos. As chamadas capturadas incluem chamadas do AWS CloudHSM console e chamadas de código para as operações AWS CloudHSM da API.

Você pode usar AWS CloudTrail para auditar qualquer chamada de API feita ao plano de AWS CloudHSM controle para garantir que nenhuma atividade indesejada esteja ocorrendo em sua conta.

Para mais detalhes, consulte [Trabalhando com AWS CloudTrail e AWS CloudHSM](#).

Monitore as CloudWatch métricas da Amazon

Você pode usar CloudWatch as métricas da Amazon para monitorar seu AWS CloudHSM cluster em tempo real. As métricas podem ser agrupadas por região, pelo ID do cluster ou pelo ID do HSM e ID do cluster.

Usando CloudWatch as métricas da Amazon, você pode configurar CloudWatch os alarmes da Amazon para alertá-lo sobre qualquer possível problema que possa afetar seu serviço. Recomendamos configurar alarmes para monitorar o seguinte:

- Aproximação do limite de chaves em um HSM
- Aproximação do limite de contagem de sessões do HSM em um HSM
- Aproximação do limite de contagem de usuários do HSM em um HSM
- Diferenças na contagem de usuários ou chaves do HSM para identificar problemas de sincronização
- Não é saudável HSMs escalar seu cluster até que AWS CloudHSM possa resolver o problema

Consulte mais detalhes em [Trabalhando com Amazon CloudWatch Logs e AWS CloudHSM Audit Logs](#).

Clusters em AWS CloudHSM

Um cluster é uma coleção de módulos de segurança de hardware (HSM) individuais que se AWS CloudHSM mantêm sincronizados. Quando você executa uma tarefa ou operação em um HSM em um cluster, o outro HSMs nesse cluster é automaticamente atualizado.

Você pode gerenciar seus AWS CloudHSM clusters a partir do [AWS CloudHSM console](#) ou de uma das [AWS SDKs ferramentas da linha de comando](#). Para obter mais informações, consulte os tópicos a seguir.

Para criar um cluster, consulte [Introdução](#).

Os tópicos a seguir fornecem informações adicionais sobre clusters.

Tópicos

- [AWS CloudHSM arquitetura de cluster](#)
- [AWS CloudHSM sincronização de clusters](#)
- [AWS CloudHSM alta disponibilidade e balanceamento de carga do cluster](#)
- [AWS CloudHSM modos de cluster](#)
- [Tipos de HSM em AWS CloudHSM](#)
- [Conecte o SDK do cliente ao cluster AWS CloudHSM](#)
- [Escalabilidade HSMs em um cluster AWS CloudHSM](#)
- [Excluindo um cluster AWS CloudHSM](#)
- [Criação de AWS CloudHSM clusters a partir de backups](#)
- [Migração do tipo de cluster do HSM](#)

AWS CloudHSM arquitetura de cluster

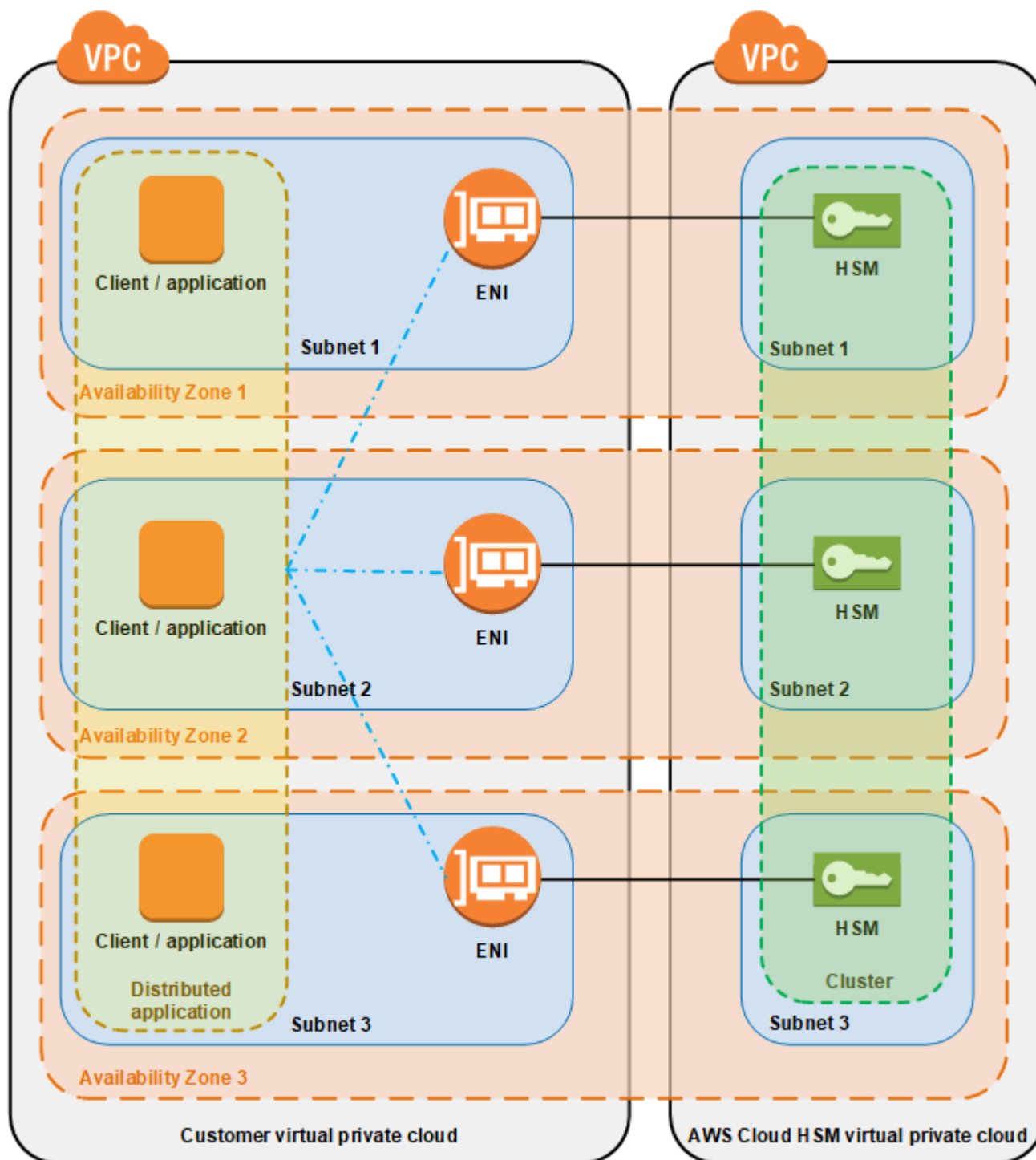
Ao criar um cluster, você especifica uma Amazon Virtual Private Cloud (VPC) em sua AWS conta e uma ou mais sub-redes nessa VPC. Recomendamos que você crie uma sub-rede em cada zona de disponibilidade (AZ) na AWS região escolhida. Você pode criar sub-redes privadas ao criar uma VPC. Para saber mais, consulte [Crie uma nuvem privada virtual \(VPC\) para AWS CloudHSM](#).

Toda vez que criar um HSM, especifique o cluster e a zona de disponibilidade para o HSM. Ao colocar o HSMs em diferentes zonas de disponibilidade, você obtém redundância e alta disponibilidade no caso de uma zona de disponibilidade não estar disponível.

Ao criar um HSM, AWS CloudHSM coloca uma interface de rede elástica (ENI) na sub-rede especificada em sua conta. A interface de rede elástica é a interface para interação com o HSM. O HSM reside em uma VPC separada em uma AWS conta de propriedade da. AWS CloudHSM O HSM, e sua interface de rede correspondente, estão na mesma zona de disponibilidade.

Para interagir com os HSMs em um cluster, você precisa do software AWS CloudHSM cliente. Normalmente, você instala o cliente em instâncias do Amazon EC2, conhecidas como instâncias de cliente, que residem na mesma VPC dos HSM ENIs, conforme mostrado na figura a seguir. Porém, isso não é tecnicamente necessário; você pode instalar o cliente em qualquer computador compatível, desde que ele possa se conectar aos ENIs HSM. O cliente se comunica com o indivíduo HSMs em seu cluster por meio de seus ENIs.

A figura a seguir representa um AWS CloudHSM cluster com três HSMs, cada um em uma zona de disponibilidade diferente na VPC.



AWS CloudHSM sincronização de clusters

Em um AWS CloudHSM cluster, AWS CloudHSM mantém as chaves do indivíduo HSMs sincronizadas. Você não precisa fazer nada para sincronizar as teclas do seu HSMs. Para manter os usuários e as políticas em cada HSM sincronizados, atualize o arquivo de configuração do AWS

CloudHSM cliente antes de [gerenciar os usuários do HSM](#). Para obter mais informações, consulte [Manter os usuários do HSM em sincronia](#).

Quando você adiciona um novo HSM a um cluster, AWS CloudHSM faz um backup de todas as chaves, usuários e políticas em um HSM existente. Depois, restaura esse backup no novo HSM. Isso mantém os dois HSMs em sincronia.

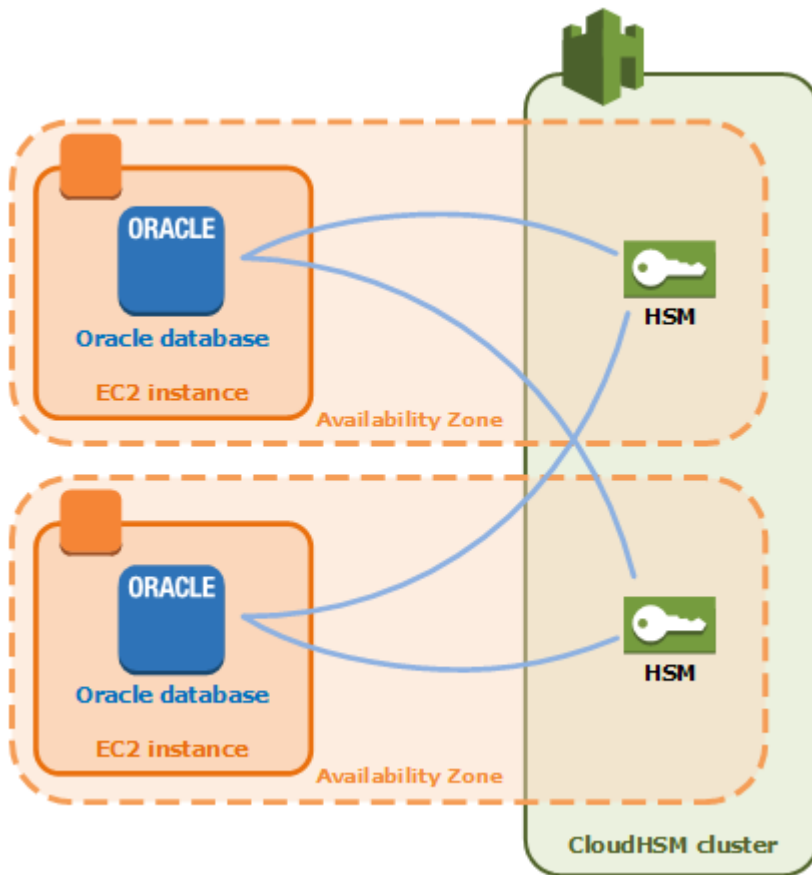
Se os HSMs em um cluster ficarem fora de sincronização, eles AWS CloudHSM serão ressincronizados automaticamente. Para habilitar isso, AWS CloudHSM usa as credenciais do usuário do [equipamento](#). Esse usuário existe em todos os HSMs serviços fornecidos por AWS CloudHSM e tem permissões limitadas. Ele pode obter um hash de objetos no HSM e extrair e inserir objetos mascarados (criptografados). O AWS não consegue visualizar nem modificar os usuários ou as chaves, além de não poder executar qualquer operação criptográfica usando essas chaves.

AWS CloudHSM alta disponibilidade e balanceamento de carga do cluster

Ao criar um AWS CloudHSM cluster com mais de um HSM, você obtém automaticamente o balanceamento de carga. O balanceamento de carga significa que o [AWS CloudHSM cliente](#) distribui as operações criptográficas em todo o cluster com base HSMs na capacidade de processamento adicional de cada HSM.

Ao criar o HSMs em diferentes zonas de AWS disponibilidade, você obtém automaticamente alta disponibilidade. Alta disponibilidade significa obter maior confiabilidade, pois nenhum HSM individual é um ponto único de falha. Recomendamos que você tenha no mínimo dois HSMs em cada cluster, com cada HSM em diferentes zonas de disponibilidade dentro de uma AWS região.

Por exemplo, a figura a seguir mostra um aplicativo de banco de dados Oracle que é distribuído para duas diferentes zonas de disponibilidade. As instâncias do banco de dados armazenam suas chaves mestras em um cluster que inclui um HSM em cada zona de disponibilidade. AWS CloudHSM sincroniza automaticamente as chaves com ambas para HSMs que elas sejam imediatamente acessíveis e redundantes.



AWS CloudHSM modos de cluster

AWS CloudHSM oferece clusters em dois modos: FIPS e não FIPS. No modo FIPS, somente chaves e algoritmos validados pelo Federal Information Processing Standard (FIPS – Padrão federal de processamento de informações) podem ser usados. O modo não FIPS oferece todas as chaves e algoritmos suportados AWS CloudHSM, independentemente da aprovação do FIPS.

Analise os detalhes nesta página antes de decidir qual modo de cluster e HSM atende às suas necessidades.

Note

Todos os clusters criados antes de 10 de junho de 2024 estão no modo FIPS e têm o tipo de HSM `hsm1.medium`.

Para conferir o modo e o tipo de HSM do seu cluster, use o comando [describe-clusters](#).

A tabela a seguir lista as principais diferenças entre cada modo de cluster:

Atributo de diferenciação	Modo do FIPS	Modo não FIPS
Compatibilidade do tipo de HSM	Disponível com os tipos hsm1.medium e hsm2m.medium.	Disponível com o tipo hsm2.medium.
Compatibilidade retroativa	Só pode ser usado para fazer backup e restauração de clusters no modo FIPS.	Só pode ser usado para fazer backup de clusters de restauração no modo não FIPS.
Seleção de chaves	Oferece suporte para a geração e o uso de chaves com mecanismos aprovados pelo FIPS ¹ .	Oferece suporte para a geração e o uso de chaves com todos os mecanismos validados pelo FIPS, além de outros mecanismos não validados.
Algoritmos	Suporta AWS CloudHSM algoritmos ¹ aprovados pelo FIPS.	Suporta AWS CloudHSM algoritmos aprovados pelo FIPS e não aprovados pelo FIPS.

[1] Consulte [Notificações de suspensão de uso](#) para obter mais detalhes.

Antes de escolher um modo de cluster, observe que o modo de um cluster (FIPS ou não FIPS) não pode ser alterado depois da sua criação, portanto, selecione o modo correto para suas necessidades.

Tipos de HSM em AWS CloudHSM

AWS CloudHSM também oferece dois tipos de módulo de segurança de hardware (HSM): hsm1.medium e hsm2m.medium. Analise os detalhes nesta página antes de decidir qual tipo de HSM atende às suas necessidades.

Além dos modos de cluster, AWS CloudHSM oferece dois tipos de HSM: hsm1.medium e hsm2m.medium. Cada tipo de HSM usa hardware diferente, e cada cluster só pode conter um tipo de HSM. A tabela a seguir lista as principais diferenças entre os dois:

Atributo de diferenciação	hsm1.medium	hsm2m.medium
Compatibilidade de modo de cluster	Disponível para clusters no modo FIPS.	Disponível para clusters no modo FIPS ou não FIPS.
Compatibilidade do tipo de rede	Indisponível	Disponível para clusters no modo FIPS ou não FIPS.
Compatibilidade retroativa	Pode ser usado para fazer backup e restaurar clusters hsm1.medium e hsm2m.medium no modo FIPS.	Só pode ser usado para fazer backup e restauração em clusters hsm2m.medium.
Capacidade de chaves	3.300 por cluster.	Total de 16.666 chaves, com chaves assimétricas com um máximo de 3.333 por cluster.
Cliente SDKs	Suporta todos os clientes SDKs.	Suporta todos os clientes SDKs.
Versões do Client SDK	Compatível com o Client SDK versão 3.1.0 e posterior.	Compatível com o SDK do cliente versão 5.9.0 e posterior.
Disponibilidade de regiões	O CloudHSM não dá mais suporte à criação de novos clusters em nenhuma região. AWS Para saber mais, consulte Notificações de suspensão de uso para obter detalhes.	Disponível na medida em Regiões da AWS que o CloudHSM está disponível.
Desempenho	Para ver o desempenho de cada tipo de HSM, consulte AWS CloudHSM informações de desempenho .	

Atributo de diferenciação	hsm1.medium	hsm2m.medium
Certificação	Compatível com FIPS 140-2, PCI DSS, PCI PIN e PCI-3DS. SOC2	Compatível com FIPS 140-3, PCI DSS, PCI PIN e PCI-3DS. SOC2

Conecte o SDK do cliente ao cluster AWS CloudHSM

Para se conectar ao cluster com o Client SDK 5 ou o Client SDK 3, você deve primeiro fazer duas coisas:

- Tenha um certificado de emissão em vigor na instância do EC2
- Inicialize o Client SDK no cluster

Coloque um certificado de emissão em cada instância do EC2

Você cria o certificado de emissão ao inicializar o cluster. Copie o certificado de emissão para o local padrão da plataforma em cada instância do EC2 que se conecta ao cluster.

Linux

```
/opt/cloudhsm/etc/customerCA.crt
```

Windows

```
C:\ProgramData\Amazon\CloudHSM\customerCA.crt
```

Especifique o local do certificado de emissão.

Com o Client SDK 5, use a ferramenta de configuração para especificar um local para o certificado de emissão.

PKCS #11 library

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --hsm-ca-cert <customerCA certificate file>
```

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --hsm-ca-cert <customerCA certificate file>
```

OpenSSL Dynamic Engine

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --hsm-ca-cert <customerCA certificate file>
```

OpenSSL Dynamic Engine Provider

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --hsm-ca-cert <customerCA certificate file>
```

Key Storage Provider (KSP)

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --hsm-ca-cert <customerCA certificate file>
```

JCE provider

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-jce --hsm-ca-cert <customerCA certificate file>
```

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --hsm-ca-cert <customerCA certificate file>
```

CloudHSM CLI

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-cli --hsm-ca-cert <customerCA certificate file>
```

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --hsm-ca-cert <customerCA certificate file>
```

Para obter mais informações, consulte [Configurar ferramenta](#).

Para obter mais informações sobre como inicializar o cluster ou criar e assinar o certificado, consulte [Iniciar o cluster](#).

Bootstrap o Client SDK

O processo de bootstrap é diferente dependendo da versão do Client SDK que você está usando, mas você deve ter o endereço IP de um dos módulos de segurança de hardware (HSM) no cluster. Você pode usar o endereço IP de qualquer HSM conectado ao seu cluster. Depois que o SDK do cliente se conecta, ele recupera os endereços IP de qualquer outro HSMs e executa o balanceamento de carga e as operações de sincronização de chaves do lado do cliente.

Para obter um endereço IP para o cluster

Para obter um endereço IP para um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. Para abrir a página de detalhes do cluster, na tabela do cluster, escolha o ID do cluster.
4. Para obter o endereço IP, vá até a HSMs guia. Para IPv4 clusters, escolha um endereço listado em IPv4 Endereço ENI. Para clusters de pilha dupla, use o endereço ENI IPv4 ou ENI. IPv6

Para obter um endereço IP para um HSM (AWS CLI)

- Obtenha o endereço IP de um HSM usando o comando [describe-clusters](#) do AWS CLI. Na saída do comando, o endereço IP do HSMs são os valores de `EniIp` e `EniIPv6` (se for um cluster de pilha dupla).

```
$ aws cloudhsmv2 describe-clusters
{
  "Clusters": [
    { ... }
    "Hsms": [
      {
...
        "EniIp": "10.0.0.9",
...
      },
      {
...
        "EniIp": "10.0.1.6",
        "EniIPv6": "2600:113f:404:be09:310e:ed34:3412:f733",
```

...

Para obter mais informações sobre ações de bootstrap, consulte [Configurar ferramenta](#).

Para ações de bootstrap no Client SDK 5

PKCS #11 library

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 -a <HSM IP addresses>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" -a <HSM IP  
addresses>
```

OpenSSL Dynamic Engine

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-dyn -a <HSM IP addresses>
```

OpenSSL Dynamic Engine Provider

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider -a <HSM IP addresses>
```

Key Storage Provider (KSP)

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" -a <HSM IP addresses>
```

JCE provider

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-jce -a <HSM IP addresses>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" -a <HSM IP addresses>
```

CloudHSM CLI

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the HSMs>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI  
IPv4 / IPv6 addresses of the HSMs>
```

Note

you can use the parameter `--cluster-id` in place of `-a <HSM_IP_ADDRESSES>`.
Para ver os requisitos de uso de `--cluster-id`, consulte [AWS CloudHSM Ferramenta de configuração do Client SDK 5](#).

Para ações de bootstrap no Client SDK 3

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 3

- Use o configure para especificar o endereço IP de um HSM no seu cluster.

```
sudo /opt/cloudhsm/bin/configure -a <IP address>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 3

- Use o configure para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM IP address>
```

Para obter mais informações sobre configuração, consulte [???](#).

Escalabilidade HSMs em um cluster AWS CloudHSM

Para aumentar ou reduzir seu AWS CloudHSM cluster, adicione ou remova HSMs usando o [AWS CloudHSM console](#) ou uma das [ferramentas da AWS SDKs linha de comando](#). Recomendamos

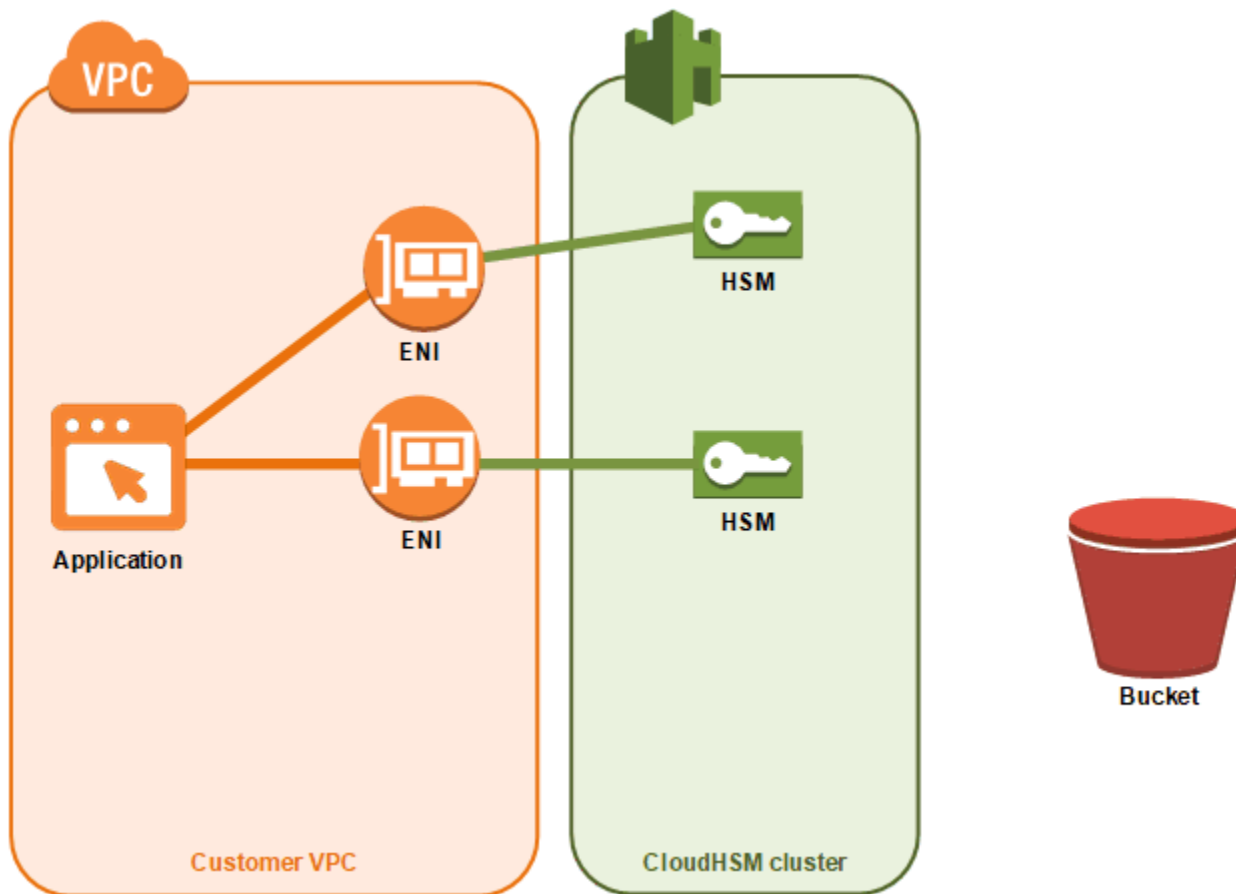
testar a carga do seu cluster para determinar a carga máxima que você deve prever e, em seguida, adicionar mais um HSM a ele para garantir a alta disponibilidade.

Tópicos

- [Adicionar um HSM a um cluster AWS CloudHSM](#)
- [Removendo um HSM de um cluster AWS CloudHSM](#)

Adicionar um HSM a um cluster AWS CloudHSM

A figura a seguir ilustra os eventos que ocorrem quando você adiciona um HSM a um cluster.



1. Você adiciona um novo HSM a um cluster. Os procedimentos a seguir explicam como fazer isso com o [console do AWS CloudHSM](#), a [AWS Command Line Interface \(AWS CLI\)](#) e a API do [AWS CloudHSM](#).

Esta é a única ação que você deve realizar. Os eventos restantes ocorrem automaticamente.

2. AWS CloudHSM faz uma cópia de backup de um HSM existente no cluster. Para obter mais informações, consulte [Backups](#).

3. AWS CloudHSM restaura o backup no novo HSM. Isso garante que o HSM esteja em sincronia com os outros no cluster.
4. O existente HSMs no cluster notifica o AWS CloudHSM cliente de que há um novo HSM no cluster.
5. O cliente estabelece uma conexão com o novo HSM.

Para adicionar um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Escolha um cluster para o HSM que você está adicionando.
3. Na HSMsguia, escolha Criar HSM.
4. Escolha uma zona de disponibilidade (AZ) para o HSM que está sendo criado. Em seguida, escolha Criar.

Para adicionar um HSM (AWS CLI)

- No prompt de comando, emita o comando [create-hsm](#), especificando um ID de cluster e uma zona de disponibilidade para o HSM que você está criando. Se você não conhece o ID do seu cluster preferido, emita o comando [describe-clusters](#). Especifique a zona de disponibilidade no formato us-east-2a, us-east-2b etc.

```
$ aws cloudhsmv2 create-hsm --cluster-id <cluster ID> --availability-  
zone <Availability Zone>  
{  
  "Hsm": {  
    "State": "CREATE_IN_PROGRESS",  
    "ClusterId": "cluster-5a73d5qzrdh",  
    "HsmId": "hsm-lgavqitns2a",  
    "SubnetId": "subnet-0e358c43",  
    "AvailabilityZone": "us-east-2c",  
    "EniId": "eni-bab18892",  
    "EniIp": "10.0.3.10",  
    "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733"  
  }  
}
```

Para adicionar um HSM (AWS CloudHSM API)

- Envie uma solicitação [CreateHsm](#), especificando o ID do cluster e uma zona de disponibilidade para o HSM que você está criando.

Removendo um HSM de um cluster AWS CloudHSM

Você pode remover um HSM usando o [AWS CloudHSM console](#) [AWS CLI](#), o ou a AWS CloudHSM API.

Para remover um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Escolha o cluster que contém o HSM que você está removendo.
3. Na HSMsguia, escolha o HSM que você está removendo. Em seguida, escolha Excluir HSM.
4. Confirme que você deseja excluir o HSM. Escolha Excluir.

Para remover um HSM (AWS CLI)

- Em um prompt de comando, emita o comando [delete-hsm](#). Transmita o ID do cluster que contém o HSM que você está excluindo e um dos seguintes identificadores de HSM:
 - O ID do HSM (`--hsm-id`)
 - O endereço IP do HSM (`--eni-ip`)
 - O ID da interface de rede elástica do HSM (`--eni-id`)

Se você não conhece os valores desses identificadores, emita o comando [describe-clusters](#).

```
$ aws cloudhsmv2 delete-hsm --cluster-id <cluster ID> --eni-ip <HSM IP address>
{
  "HsmId": "hsm-lgavqitns2a"
}
```

Para remover um HSM (AWS CloudHSM API)

- Envie uma solicitação [DeleteHsm](#), especificando o ID do cluster e um identificador para o HSM que você está excluindo.

Excluindo um cluster AWS CloudHSM

Antes de excluir um cluster, você deve remover tudo HSMs do cluster. Para obter mais informações, consulte [Removendo um HSM de um cluster AWS CloudHSM](#).

Depois de remover tudo HSMs, você pode excluir um cluster usando o [AWS CloudHSM console](#), o [AWS Command Line Interface \(AWS CLI\)](#) ou a AWS CloudHSM API.

Para excluir um cluster (console)

- Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
- Escolha o cluster que você está excluindo. Em seguida, escolha Delete cluster.
- Confirme que deseja excluir o cluster e escolha Delete.

Para excluir um cluster (AWS CLI)

- No prompt de comando, emita o comando [delete-cluster](#), transmitindo o ID do cluster que você está excluindo. Se você não conhece o ID do cluster, emita o comando [describe-clusters](#).

```
$ aws cloudhsmv2 delete-cluster --cluster-id <cluster ID>
{
  "Cluster": {
    "Certificates": {
      "ClusterCertificate": "<certificate string>"
    },
    "SourceBackupId": "backup-rtq2dwi2gq6",
    "SecurityGroup": "sg-40399d28",
    "CreateTimestamp": 1504903546.035,
    "SubnetMapping": {
      "us-east-2a": "subnet-f1d6e798",
      "us-east-2c": "subnet-0e358c43",
      "us-east-2b": "subnet-40ed9d3b"
    },
    "ClusterId": "cluster-kdmrayrc7gi",
    "VpcId": "vpc-641d3c0d",
```

```
    "State": "DELETE_IN_PROGRESS",
    "HsmType": "hsm1.medium",
    "StateMessage": "The cluster is being deleted.",
    "Hsms": [],
    "BackupPolicy": "DEFAULT"
  }
}
```

Para excluir um cluster do AWS CloudHSM (API)

- Envie uma solicitação [DeleteCluster](#), especificando o ID do cluster que você está excluindo.

Criação de AWS CloudHSM clusters a partir de backups

Para restaurar um AWS CloudHSM cluster a partir de um backup, siga as etapas deste tópico. Seu cluster contém os mesmos usuários, material de chave, certificados, configuração e políticas que estavam no backup restaurado. Para obter mais informações sobre gerenciar backups, consulte [Backups do cluster](#).

Criar clusters de backups (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione Create cluster (Criar cluster).
3. Na seção Configuração de cluster, faça o seguinte:
 - a. Para VPC, escolha uma VPC para o cluster que você está criando.
 - b. Para AZ(s), escolha uma sub-rede privada para cada zona de disponibilidade que você está adicionando ao cluster.
 - c. Em Tipo de rede, escolha o protocolo IP HSMs que você usará para conexões.
4. Na seção Cluster source, faça o seguinte:
 - a. Escolha Restore cluster from existing backup.
 - b. Escolha o backup que você está restaurando.
5. Escolha Próximo: revisar.
6. Reveja a sua configuração de cluster e escolha Create cluster.
7. Especifique por quanto tempo o serviço deve reter os backups.

Aceite o período de retenção padrão de 90 dias ou digite um novo valor entre 7 e 379 dias. O serviço excluirá automaticamente os backups presentes nesse cluster que sejam mais antigos do que o valor especificado. Você pode alterar esse valor depois. Para obter mais informações, consulte [Configurar a retenção de backup](#).

8. Escolha Próximo.
9. (Opcional) Digite uma chave de tag e um valor de tag opcional. Para adicionar mais de uma tag ao cluster, selecione Adicionar tag.
10. Escolha Review (Revisar).
11. Reveja sua configuração de cluster e escolha Create cluster (Criar cluster).

Tip

Para criar um HSM que contenha os mesmos usuários, material de chave, certificados, configuração e políticas que estavam no backup restaurado, [adicione um HSM](#) ao cluster.

Criar clusters de backups (AWS CLI)

Para determinar o ID de backup, emita o comando [describe-backups](#).

- Em um prompt de comando, emita o comando [create-cluster](#). Especifique o tipo de instância do HSM, a sub-rede IDs das sub-redes em que você planeja criar HSMs e a ID de backup do backup que você está restaurando.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
                                --subnet-ids <subnet ID 1> <subnet ID 2> <subnet ID
N> \
                                --source-backup-id <backup ID>
                                --mode <FIPS> \
                                --network-type <IPV4>
{
  "Cluster": {
    "HsmType": "hsm2m.medium",
    "VpcId": "vpc-641d3c0d",
    "Hsms": [],
    "State": "CREATE_IN_PROGRESS",
    "SourceBackupId": "backup-rtq2dwi2gq6",
```

```
"BackupPolicy": "DEFAULT",
"BackupRetentionPolicy": {
  "Type": "DAYS",
  "Value": 90
},
"NetworkType": "IPv4",
"SecurityGroup": "sg-640fab0c",
"CreateTimestamp": 1504907311.112,
"SubnetMapping": {
  "us-east-2c": "subnet-0e358c43",
  "us-east-2a": "subnet-f1d6e798",
  "us-east-2b": "subnet-40ed9d3b"
},
"Certificates": {
  "ClusterCertificate": "<certificate string>"
},
"ClusterId": "cluster-jxhlf7644ne"
}
```

Crie clusters a partir de backups (AWS CloudHSM API)

Consulte o tópico a seguir para saber como criar clusters a partir de backups usando a API.

- [CreateCluster](#)

Migração do tipo de cluster do HSM

AWS CloudHSM oferece a capacidade de alterar o tipo de HSM de um cluster existente. Consulte a tabela nesta página para saber se a modificação do tipo de HSM é permitida.

Para obter mais informações sobre os tipos de HSMs suporte e seus recursos, consulte [Tipos de HSM em AWS CloudHSM](#).

Note

Você não pode alterar o modo FIPS de um cluster durante essa operação.

De	Para	Comment
hsm1. médio	hsm2m.medium	Permitido
hsm2m.medium	hsm1. médio	Condicional. Você pode reverter de hsm2m.medium para hsm1.medium dentro de 24 horas após o início de uma migração.

Tópicos

- [Migrar de hsm1.medium para hsm2m.medium](#)

Migrar de hsm1.medium para hsm2m.medium

Você pode migrar seu AWS CloudHSM cluster de hsm1.medium para hsm2m.medium. Este tópico descreve os pré-requisitos, o processo de migração e os procedimentos de reversão.

Antes de iniciar a migração, certifique-se de que sua aplicação siga as recomendações em [Arquitete seu cluster para alta disponibilidade](#). Isso ajuda a evitar o tempo de inatividade durante o processo.

Note

As migrações automáticas para hsm2m.medium começarão em 20 de janeiro de 2026.

Visão geral do processo de migração de hsm1.medium para hsm2m.medium

Você pode iniciar a migração usando o AWS CloudHSM console AWS CLI, o ou a AWS CloudHSM API. Não importa onde você a inicie, a migração do AWS CloudHSM cluster usa o endpoint da `modify-cluster` API. Como alternativa, o AWS CloudHSM migrará automaticamente o cluster em seu nome. Quando a migração começa, todo o cluster entra em um modo de gravação limitada. Para obter mais informações, consulte [Modo de gravação limitada de cluster](#).

Para minimizar o impacto, AWS CloudHSM HSMs muda de hsm1.medium para hsm2m.medium, um por vez. A substituição HSMs mantém os mesmos endereços IP, não exigindo, portanto, nenhuma alteração na configuração durante ou após a migração.

Como funciona a migração:

1. Antes de migrar o primeiro HSM, AWS CloudHSM crie um backup completo de todo o cluster.
2. Usando esse backup, AWS CloudHSM cria um novo HSM do tipo solicitado (hsm2m.medium) para substituir o primeiro HSM.
3. Antes de migrar cada HSM subsequente, AWS CloudHSM cria um novo backup completo de todo o cluster.
4. AWS CloudHSM repete as etapas 3 e 4 para cada HSM no cluster, migrando um HSM por vez.
5. Cada migração individual de HSM leva aproximadamente 30 minutos.

AWS CloudHSM monitora a integridade do cluster e realiza validações em todo o processo de migração. Se AWS CloudHSM detectar um aumento nos erros ou uma verificação de validação falhar, ela interromperá automaticamente a migração e reverterá o cluster para seu tipo de HSM original. Você também pode reverter manualmente por até 24 horas após o início da migração. Antes de reverter, consulte [Considerações sobre a reversão do tipo de HSM](#).

Pré-requisitos para migrar para hsm2m.medium

Seu AWS CloudHSM cluster existente deve atender a esses requisitos para migrar para hsm2m.medium. Se alguma condição não for atendida durante as verificações de validação, o AWS CloudHSM reverte automaticamente o cluster para seu tipo de HSM original.

Para obter uma lista de problemas de migração conhecidos, consulte [???](#)

- Nos últimos 7 dias:
 - Todas as conexões de clientes usaram o SDK 5.9 ou superior.
 - Ao executar o ECDSA Verify, todas as conexões do cliente usaram o SDK 5.13 ou superior.
 - AWS CloudHSM as instâncias usaram apenas funcionalidades suportadas (e nenhuma das obsoletas). Consulte [Notificações de suspensão de uso](#) para obter mais detalhes.
 - Você deve ter usado um SDK para se conectar a pelo menos um HSM no cluster nos últimos 7 dias.
- O cluster tem o estado ACTIVE.
- O cluster tem 27 HSMs ou menos.
- A taxa de erro das operações do HSM não aumenta durante a migração.

 Note

A restrição anterior que impedia a migração de clientes com workloads de chaves de token foi removida.

Modo de gravação limitada no cluster

Quando seu cluster inicia a migração, ele entra em um modo de gravação limitada. As operações que podem alterar o estado do HSM são rejeitadas. Todas as operações de leitura permanecem inalteradas.

Durante a migração, sua aplicação recebe um erro do HSM ao tentar estas operações:

- Geração e exclusão de chaves de token (as workloads da chave de sessão continuam operando).
- Todas as criações, exclusões ou modificações de usuários.
- Operações de quórum.
- Modificação de chaves no HSM, como alteração de atributos de chave.
- Registro de mTLS.

AWS CloudHSM também coloca seu cluster em um MODIFY_IN_PROGRESS estado durante a migração. Durante esse período, você não pode adicionar ou remover HSMs do cluster.

Iniciar a migração

O processo de migração do cluster substitui um indivíduo HSMs em seu cluster, um por vez. A duração depende do número de HSMs no seu cluster. Em média, esse processo leva cerca de 30 minutos por HSM. Você pode acompanhar o progresso monitorando o tipo de indivíduo do HSM HSMs no cluster para ver quantos foram migrados para o novo tipo.

Console

Para alterar o tipo de HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione o botão de seleção ao lado do ID do cluster que você deseja alterar
3. No menu Ações, escolha Modify HSM Type e selecione o tipo de HSM desejado.

Esse procedimento coloca seu cluster no estado MODIFY_IN_PROGRESS. Após a migração, seu cluster retorna ao estado ACTIVE.

AWS CLI

Para alterar o tipo de HSM ([AWS CLI](#))

- Em um prompt de comando, execute o comando [modify-cluster](#). Especifique o ID do cluster e o tipo de HSM desejado.

```
$ aws cloudhsmv2 modify-cluster --cluster-id <cluster ID> --hsm-type <HSM Type>

{
  "Cluster": {
    "BackupPolicy": "DEFAULT",
    "BackupRetentionPolicy": {
      "Type": "DAYS",
      "Value": 90
    },
    "VpcId": "vpc-50ae0636",
    "SubnetMapping": {
      "us-west-2b": "subnet-49a1bc00",
      "us-west-2c": "subnet-6f950334",
      "us-west-2a": "subnet-fd54af9b"
    },
    "SecurityGroup": "sg-6cb2c216",
    "HsmType": "hsm2m.medium",
    "HsmTypeRollbackExpiration": 1730383180.000,
    "Certificates": {},
    "State": "MODIFY_IN_PROGRESS",
    "Hsms": [],
    "ClusterId": "cluster-igklspoyj5v",
    "ClusterMode": "FIPS",
    "CreateTimestamp": 1502423370.069
  }
}
```

Esse procedimento coloca seu cluster no estado MODIFY_IN_PROGRESS. Após a migração, seu cluster retorna ao estado ACTIVE.

AWS CloudHSM API

Para alterar o tipo de HSM (AWS CloudHSM API)

- Envie uma solicitação [ModifyCluster](#). Especifique o ID do cluster e o tipo de HSM desejado para o cluster.

Esse procedimento coloca seu cluster no estado `MODIFY_IN_PROGRESS`. Após a migração, seu cluster retorna ao estado `ACTIVE`.

Reverter a migração

AWS CloudHSM monitora taxas de erro elevadas e executa verificações de validação contínuas durante toda a migração. Se AWS CloudHSM detectar uma diminuição na qualidade do serviço ou qualquer falha na validação, ele iniciará automaticamente uma reversão para o tipo de HSM original do seu cluster. Durante uma reversão, para cada HSM no cluster:

- AWS CloudHSM usa o backup feito no início da migração desse HSM.
- Ele substitui um HSM por vez até que todos HSMs retornem ao tipo original.
- Seu cluster permanece no modo de gravação limitada durante todo o processo.

Você pode reverter a migração dentro de 24 horas após iniciá-la. Para verificar o prazo de reversão:

1. Execute o comando [describe-clusters](#).
2. Procure o valor de `HsmTypeRollbackExpiration`. Esse carimbo de data/hora é seu prazo de reversão.

Se você decidir reverter, faça isso antes desse prazo. A reversão usa o backup mais recente do seu tipo de HSM original.

Warning

Tenha cuidado ao reverter após a conclusão da migração. Se você concluir uma migração e depois usá-la AWS CloudHSM para criar novas chaves ou usuários, a reversão pode resultar em perda de dados. Consulte [Sincronizar dados após uma reversão](#) para saber como mitigar a perda de dados após uma reversão.

Console

Para reverter seu tipo de HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Selecione o ID do cluster que deseja reverter.
3. No menu Ações, escolha Modify HSM Type e selecione o tipo de HSM original.

Esse procedimento coloca seu cluster no estado ROLLBACK_IN_PROGRESS. Após a reversão, seu cluster retorna ao estado ACTIVE.

AWS CLI

Para reverter seu tipo de HSM ([AWS CLI](#))

- Em um prompt de comando, execute o comando [modify-cluster](#). Especifique o ID do cluster e o tipo de HSM original.

```
$ aws cloudhsmv2 modify-cluster --cluster-id <cluster ID> --hsm-type <HSM Type>

{
  "Cluster": {
    "BackupPolicy": "DEFAULT",
    "BackupRetentionPolicy": {
      "Type": "DAYS",
      "Value": 90
    },
    "VpcId": "vpc-50ae0636",
    "SubnetMapping": {
      "us-west-2b": "subnet-49a1bc00",
      "us-west-2c": "subnet-6f950334",
      "us-west-2a": "subnet-fd54af9b"
    },
    "SecurityGroup": "sg-6cb2c216",
    "HsmType": "hsm1.medium",
    "HsmTypeRollbackExpiration": 1730383180.000,
    "Certificates": {},
    "State": "ROLLBACK_IN_PROGRESS",
    "Hsms": [],
    "ClusterId": "cluster-igklspoyj5v",
    "ClusterMode": "FIPS",
    "CreateTimestamp": 1502423370.069
  }
}
```

```
}  
}
```

Esse procedimento coloca seu cluster no estado `ROLLBACK_IN_PROGRESS`. Após a reversão, seu cluster retorna ao estado `ACTIVE`.

AWS CloudHSM API

Para reverter seu tipo de HSM (AWS CloudHSM API)

- Envie uma solicitação [ModifyCluster](#). Especifique o ID do cluster e o tipo de HSM original para o cluster.

Esse procedimento coloca seu cluster no estado `ROLLBACK_IN_PROGRESS`. Após a reversão, seu cluster retorna ao estado `ACTIVE`.

Sincronizar dados após uma reversão

Durante a migração, HSMs estão no modo de gravação limitada, evitando alterações no estado do HSM. Se você reverter durante esse período (enquanto o cluster estiver `MODIFY_IN_PROGRESS`), isso resultará em um cluster com conteúdo idêntico ao cluster original.

Depois que seu cluster retornar ao estado `ACTIVE`, o modo de gravação limitada será suspenso. Se você criar uma chave ou usuário enquanto estiver no estado `ACTIVE` e depois reverter, essa chave ou usuário não estará presente no seu cluster revertido.

Para abordar a sincronização de usuários, use o [gerenciamento de usuários](#) com a CloudHSM CLI para recriar os usuários ausentes em seu cluster revertido. Os usuários devem ser recriados manualmente porque o comando `user replicate` não oferece suporte à sincronização de usuários de `hsm2m.medium` para `hsm1.medium`. Veja os [problemas conhecidos](#) de replicação de usuários.

Para abordar a sincronização de chaves, use o comando [key replicate](#) para replicar uma chave entre dois clusters. Se não tiver instalado a CloudHSM CLI, siga as instruções em [???](#).

Para sincronizar as chaves após a reversão

Siga estas etapas após concluir a reversão. Usaremos estes termos:

- “cluster-1”: seu cluster revertido (agora hsm1.medium)
- “cluster-2”: um novo cluster hsm2m.medium temporário que você criará

1. Crie um novo cluster hsm2m.medium (cluster-2) usando o backup hsm2m.medium mais recente do cluster-1:

```
aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \  
                                --subnet-ids <subnet ID 1> <subnet ID 2> <subnet ID  
N> \  
                                --source-backup-id <backup ID>  
                                --mode <FIPS>
```

2. Crie um HSM no cluster-2:

```
aws cloudhsmv2 create-hsm --cluster-id <cluster-2 ID>
```

3. Liste as chaves no cluster-2 que precisam de replicação:

```
cloudhsm-cli key list --cluster-id <cluster-2 ID>
```

4. Replique cada chave do cluster-2 para o cluster-1:

```
cloudhsm-cli key replicate --source-cluster-id <cluster-2 ID> \  
                            --destination-cluster-id <cluster-1 ID> \  
                            --filter attr.label=<key ID>
```

5. Repita a etapa 4 para cada chave que precisa ser copiada.
6. Exclua o HSM no cluster-2:

```
aws cloudhsmv2 delete-hsm --cluster-id <cluster-2 ID> --hsm-id <HSM ID>
```

7. Exclua o cluster-2:

```
aws cloudhsmv2 delete-cluster --cluster-id <cluster-2 ID>
```

Usuários do HSM em AWS CloudHSM

Antes de usar seu AWS CloudHSM cluster para processamento de criptografia, você deve criar usuários e [chaves](#) nos módulos de segurança de hardware (HSM) em seu cluster.

Note

Os usuários do HSM são diferentes dos usuários do IAM. Os usuários do IAM que têm as credenciais corretas podem criar interagindo com recursos HSMs por meio da API da AWS. Depois que o HSM for criado, você deverá usar as credenciais de usuário do HSM para autenticar as operações no HSM.

Em AWS CloudHSM, você deve usar as ferramentas de linha de comando da [CLI do CloudHSM ou do CloudHSM Management Utility \(CMU\)](#) para criar e gerenciar os usuários no seu HSM. [A CLI do CloudHSM foi projetada para ser usada com a série de versões mais recentes do SDK, enquanto a CMU foi projetada para ser usada com a série de versões anteriores do SDK.](#)

Consulte os seguintes tópicos para obter mais informações sobre como gerenciar usuários do HSM no AWS CloudHSM. Você também pode aprender a usar a autenticação de quorum (também conhecida como controle de acesso M de N).

Tópicos

- [Gerenciamento de usuários do HSM com a CLI do CloudHSM](#)
- [Gerenciamento de usuários do HSM com o CloudHSM Management Utility \(CMU\)](#)

Gerenciamento de usuários do HSM com a CLI do CloudHSM

[Para gerenciar os usuários do módulo de segurança de hardware \(HSM\) AWS CloudHSM, você deve fazer login no HSM com o nome de usuário e a senha de um administrador.](#) Somente administradores podem gerenciar usuários. O HSM contém um admin padrão chamado admin. Você define a senha para admin quando [ativou o cluster](#).

Este tópico fornece step-by-step instruções e detalhes sobre o gerenciamento de usuários do HSM com a CLI do CloudHSM.

Tópicos

- [Pré-requisitos para gerenciamento de usuários na CLI do CloudHSM](#)
- [Tipos de usuários do HSM para a CLI do CloudHSM](#)
- [Tabela de permissões de usuário do HSM para a CLI do CloudHSM](#)
- [Criar um administrador de usuários do HSM usando a CLI do CloudHSM](#)
- [Criar um usuário criptográfico do HSM usando a CLI do CloudHSM](#)
- [Listar todos os usuários do HSM no cluster usando a CLI do CloudHSM](#)
- [Alterar as senhas de usuário do HSM usando a CLI do CloudHSM](#)
- [Excluir usuários do HSM usando a CLI do CloudHSM](#)
- [Gerenciar a MFA para usuários do HSM usando a CLI do CloudHSM](#)
- [Gerenciar a autenticação de quórum \(controle de acesso M ou N\) usando a CLI do CloudHSM](#)

Pré-requisitos para gerenciamento de usuários na CLI do CloudHSM

Antes de usar a CLI do CloudHSM para gerenciar AWS CloudHSM usuários de módulos de segurança de hardware (HSM) no, você deve preencher esses pré-requisitos. Os tópicos a seguir descrevem como começar a usar a CLI do CloudHSM.

Tópicos

- [Obtenha o endereço IP de um HSM em AWS CloudHSM](#)
- [Fazer o download da CLI do CloudHSM](#)

Obtenha o endereço IP de um HSM em AWS CloudHSM

Para usar a CLI do CloudHSM, você deve usar a ferramenta de configuração para atualizar a configuração local. Para obter instruções sobre como executar a ferramenta de configuração com a CLI do CloudHSM, consulte. [Introdução à Interface de Linha de AWS CloudHSM Comando \(CLI\)](#) O parâmetro -a exige que você adicione o endereço IP de um HSM no seu cluster. Se você tiver vários HSMs, poderá usar qualquer endereço IP. Isso garante que a CLI do CloudHSM possa propagar todas as alterações que você fizer em todo o cluster. Lembre-se de que a CLI do CloudHSM usa seu arquivo local para rastrear as informações do cluster. Se o cluster mudou desde a última vez em que você usou o CloudHSM CLI de um host específico, você deve adicionar essas alterações ao arquivo de configuração local armazenado nesse host. Nunca remova um HSM enquanto estiver usando a CLI do CloudHSM.

Para obter um endereço IP para um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. Para abrir a página de detalhes do cluster, na tabela do cluster, escolha o ID do cluster.
4. Para obter o endereço IP, vá até a HSMs guia. Para IPv4 clusters, escolha um endereço listado em IPv4 Endereço ENI. Para clusters de pilha dupla, use o endereço ENI IPv4 ou ENI. IPv6

Para obter um endereço IP para um HSM (AWS CLI)

- Obtenha o endereço IP de um HSM usando o comando [describe-clusters](#) do AWS CLI. Na saída do comando, o endereço IP do HSMs são os valores de `EniIp` e `EniIPv6` (se for um cluster de pilha dupla).

```
$ aws cloudhsmv2 describe-clusters
{
  "Clusters": [
    { ... }
    "Hsms": [
      {
...
        "EniIp": "10.0.0.9",
...
      },
      {
...
        "EniIp": "10.0.1.6",
        "EniIPv6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
    }
```

Fazer o download da CLI do CloudHSM

A versão mais recente da CLI do CloudHSM está disponível para tarefas de gerenciamento de usuários do HSM para o Client SDK 5. Para baixar e instalar a CLI do CloudHSM, siga as instruções em [Instalar e configurar a CLI do CloudHSM](#).

Tipos de usuários do HSM para a CLI do CloudHSM

A maioria das operações que você executa no módulo de segurança de hardware (HSM) exige as credenciais de um usuário do AWS CloudHSM HSM. O HSM autentica cada usuário do HSM e cada usuário do HSM tem um tipo que determina quais operações você pode realizar no HSM como esse usuário.

Note

Os usuários do HSM são diferentes dos usuários do IAM. Os usuários do IAM que têm as credenciais corretas podem criar interagindo com recursos HSMs por meio da API da AWS. Depois que o HSM for criado, você deverá usar as credenciais de usuário do HSM para autenticar as operações no HSM.

Tipos de usuário

- [Administrador desativado](#)
- [Administrador](#)
- [Crypto user \(CU\) \(usuário de criptografia\)](#)
- [Usuário de dispositivo \(AU\)](#)

Administrador desativado

Na CLI do CloudHSM, o administrador não ativado é um usuário temporário que existe somente no primeiro HSM AWS CloudHSM em um cluster que nunca foi ativado. Use o comando [na CLI do CloudHSM para](#) `cluster activate` Depois de executar esse comando, o administrador não ativado é solicitado a alterar a senha. Depois de alterar a senha, o administrador não ativado se torna administrador.

Administrador

Na CLI do CloudHSM, o administrador pode realizar operações de gerenciamento de usuários. Por exemplo, eles podem criar e excluir usuários e alterar suas senhas. Para obter mais informações sobre administradores, consulte o [Tabela de permissões de usuário do HSM para a CLI do CloudHSM](#).

Crypto user (CU) (usuário de criptografia)

Um usuário de criptografia (CU) pode realizar as seguintes operações de criptografia e gerenciamento de chaves.

- Gerenciamento de chaves: criar, excluir, compartilhar, importar e exportar chaves criptográficas.
- Operações criptográficas: use chaves criptográficas para criptografia, descriptografia, assinatura, verificação e muito mais.

Para obter mais informações, consulte o [Tabela de permissões de usuário do HSM para a CLI do CloudHSM](#).

Usuário de dispositivo (AU)

O usuário do equipamento (AU) pode realizar operações de clonagem e sincronização no seu cluster. HSMs AWS CloudHSM usa a AU para sincronizar o HSMs em um AWS CloudHSM cluster. O AU existe em todos os HSMs fornecidos por AWS CloudHSM, e tem permissões limitadas. Para obter mais informações, consulte o [Tabela de permissões de usuário do HSM para a CLI do CloudHSM](#).

AWS não pode realizar nenhuma operação no seu HSMs . AWS não pode visualizar ou modificar seus usuários ou chaves e não pode realizar nenhuma operação criptográfica usando essas chaves.

Tabela de permissões de usuário do HSM para a CLI do CloudHSM

A tabela a seguir lista as operações do HSM classificadas pelo tipo de usuário ou sessão do HSM que pode realizar a operação no AWS CloudHSM.

	Administrador	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Obter informações ¹ básicas do cluster	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)

	Administrador	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Alterar a própria senha	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	Não aplicável
Alterar a senha de qualquer usuário	 Sim	 No (Não)	 Não	 Não
Adicionar , remover usuários	 Sim	 No (Não)	 Não	 Não
Obter status ³ de sincronização	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Extrair, inserir objetos mascarados ⁴	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Funções de gerenciamento de chaves ⁵	 No (Não)	 Sim	 No (Não)	 Não
Criptografar, descriptografar	 No (Não)	 Sim	 No (Não)	 Não

	Administrador	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Assinar, verificar	 No (Não)	 Sim	 No (Não)	 Não
Gere resumos e HMACs	 No (Não)	 Sim	 No (Não)	 Não

- [1] As informações básicas do cluster incluem o número de pessoas HSMs no cluster e o endereço IP, modelo, número de série, ID do dispositivo, ID do firmware, etc.
- [2] O usuário pode obter um conjunto de arquivos de resumo (hashes) que correspondem às chaves no HSM. Um aplicativo pode comparar esses conjuntos de resumos para entender o status de sincronização HSMs em um cluster.
- [3] Objetos mascarados são chaves criptografadas antes de sair do HSM. Elas não podem ser descriptografadas fora do HSM. Somente são descriptografadas depois de serem inseridas em um HSM que esteja no mesmo cluster que o HSM do qual foram extraídas. Um aplicativo pode extrair e inserir objetos mascarados para sincronizá-los HSMs em um cluster.
- [4] As funções de gerenciamento de chaves incluem criar, excluir, encapsular, desencapsular e modificar os atributos das chaves.

Criar um administrador de usuários do HSM usando a CLI do CloudHSM

Siga essas etapas para criar um usuário administrador do HSM usando a CLI do CloudHSM.

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Use o comando login e faça login no cluster como administrador.

```
aws-cloudhsm > login --username <username> --role admin
```

3. O sistema solicita que você forneça sua senha. Você insere a senha e a saída mostra que o comando foi bem-sucedido.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "admin"
  }
}
```

4. Insira o seguinte comando para criar um administrador:

```
aws-cloudhsm > user create --username <username> --role admin
```

5. Digite a senha do novo usuário.
6. Digite a senha novamente para confirmar se a senha digitada está correta.

Criar um usuário criptográfico do HSM usando a CLI do CloudHSM

Siga estas etapas para criar um usuário de criptografia do HSM usando a CLI do CloudHSM.

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Use o comando login e faça login no cluster como administrador.

```
aws-cloudhsm > login --username <username> --role admin
```

3. O sistema solicita que você forneça sua senha. Você insere a senha e a saída mostra que o comando foi bem-sucedido.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<USERNAME>",
    "role": "admin"
  }
}
```

4. Digite o comando a seguir para criar um novo usuário de criptografia:

```
aws-cloudhsm > user create --username <username> --role crypto-user
```

5. Digite a senha do novo usuário de criptografia.
6. Digite a senha novamente para confirmar se a senha digitada está correta.

Listar todos os usuários do HSM no cluster usando a CLI do CloudHSM

Use o comando `user list` na CLI do CloudHSM para listar todos os usuários no cluster do AWS CloudHSM. Você não precisa fazer login para executar `user list`. Todos os tipos de usuário podem listar usuários.

Siga estas etapas para listar todos os usuários no cluster

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Insira o comando a seguir para listar todos os usuários no cluster:

```
aws-cloudhsm > user list
```

Para obter mais informações sobre user list, consulte [lista de usuários](#).

Alterar as senhas de usuário do HSM usando a CLI do CloudHSM

Use o comando user change-password na CLI do CloudHSM para alterar a senha de um usuário do HSM.

Os tipos de usuários e as senhas diferenciam maiúsculas e minúsculas, mas os nomes de usuários não.

Admin, os usuários de criptografia (CUs) e os usuários do dispositivo (AUs) podem alterar suas próprias senhas. Para alterar a senha de outro usuário, você deve fazer login como administrador. Não é possível alterar a senha de um usuário que está conectado no momento.

Para alterar sua senha

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Use o login comando e faça login como usuário com a senha que você deseja alterar.

```
aws-cloudhsm > login --username <username> --role <role>
```

3. Digite a senha do usuário.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "<role>"
  }
}
```

4. Digite o comando user change-password.

```
aws-cloudhsm > user change-password --username <username> --role <role>
```

5. Digite a nova senha.
6. Digite novamente a nova senha.

Para alterar a senha de outro usuário

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CloudHSM CLI, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
```

```
    "role": "admin"
  }
}
```

3. Digite o user change-password comando junto com o nome de usuário do usuário cuja senha você deseja alterar.

```
aws-cloudhsm > user change-password --username <username> --role <role>
```

4. Digite a nova senha.
5. Digite novamente a nova senha.

Para obter mais informações sobre a exclusão user change-password, consulte [troca de senha de usuário](#).

Excluir usuários do HSM usando a CLI do CloudHSM

Use user delete na CLI do CloudHSM para excluir um usuário do HSM. Você deve fazer login como administrador para excluir outro usuário.

Tip

Você não pode excluir usuários de criptografia (CU) que possuem chaves.

Para excluir um usuário

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Use o comando login e faça login no cluster como administrador.

```
aws-cloudhsm > login --username <username> --role admin
```

3. O sistema solicita que você forneça sua senha. Você insere a senha e a saída mostra que o comando foi bem-sucedido.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "admin"
  }
}
```

4. Use o comando `user delete` para excluir um usuário.

```
aws-cloudhsm > user delete --username <username> --role <role>
```

Para obter mais informações sobre `user delete`, consulte [deleteUser](#).

Gerenciar a MFA para usuários do HSM usando a CLI do CloudHSM

Para aumentar a segurança, você pode configurar a autenticação multifator (MFA) para que os usuários ajudem a proteger AWS CloudHSM o cluster.

Ao fazer login em um cluster com uma conta de usuário do HSM habilitada para MFA, você fornece sua senha à CLI do CloudHSM, o primeiro fator, o que você conhece, e a CLI do CloudHSM fornece um token e solicita que você assine o token.

Para fornecer o segundo fator, o que você tem, você assina o token com uma chave privada de um par de chaves que já criou e associou ao usuário do HSM. Para acessar o cluster, você fornece o token assinado à CLI do CloudHSM.

Para obter mais informações sobre como configurar a MFA para um usuário, consulte [Configurar o MFA para a CLI do CloudHSM](#)

Os tópicos a seguir fornecem mais informações sobre como trabalhar com a autenticação de quorum no AWS CloudHSM.

Tópicos

- [Autenticação de quorum e MFA em clusters AWS CloudHSM usando o CloudHSM CLI](#)
- [Requisitos de pares de chaves de MFA para usar a CLI do AWS CloudHSM CloudHSM](#)
- [Configurar o MFA para a CLI do CloudHSM](#)
- [Crie usuários com MFA habilitado para a CLI do CloudHSM](#)
- [Faça login com os usuários com MFA habilitado para a CLI do CloudHSM](#)
- [Rotacione as chaves para usuários com MFA habilitado para CLI do CloudHSM](#)
- [Cancelar o registro de uma chave pública de MFA usando a CLI do CloudHSM](#)
- [Referência de arquivo de token para MFA com CLI do CloudHSM](#)

Autenticação de quorum e MFA em clusters AWS CloudHSM usando o CloudHSM CLI

O AWS CloudHSM cluster usa a mesma chave para autenticação de quórum e para autenticação multifator (MFA). Isso significa que um usuário com MFA habilitado está efetivamente registrado para controle de acesso MoFN ou quórum. Para usar com sucesso a autenticação MFA e de quórum para o mesmo usuário do HSM, considere os seguintes pontos:

- Se estiver usando a autenticação de quórum para um usuário hoje, você deve usar o mesmo par de chaves que criou para o usuário do quórum para habilitar a MFA para ele.
- Se você adicionar o requisito de MFA para um usuário não MFA que não seja um usuário de autenticação de quórum, registre esse usuário como um quórum (MoFN) com autenticação MFA.
- Se você remover o requisito de MFA ou alterar a senha de um usuário de MFA que também seja usuário de autenticação de quórum, também removerá o registro do usuário do quórum como usuário do Quorum (MoFN).
- Se você remover o requisito de MFA ou alterar a senha de um usuário de MFA que também é usuário de autenticação de quórum, mas ainda quiser que esse usuário participe da autenticação de quórum, deverá registrá-lo novamente como usuário do MoFN.

Para obter mais informações sobre a autenticação de quórum, consulte [Gerenciar autenticação de quórum \(M de N\)](#).

Requisitos de pares de chaves de MFA para usar a CLI do AWS CloudHSM CloudHSM

Para habilitar a autenticação multifator (MFA) para um usuário do módulo de segurança de hardware (HSM) AWS CloudHSM em, você pode criar um novo par de chaves ou usar uma chave existente que atenda aos seguintes requisitos:

- Tipo de chave: assimétrico
- Uso da chave: assinar e verificar
- Especificação da chave: RSA_2048
- O algoritmo de assinatura inclui: SHA256with RSAEncryption

Note

Se você estiver usando a autenticação de quórum ou planeja usar a autenticação de quórum, consulte [Autenticação de quorum e MFA em clusters AWS CloudHSM usando o CloudHSM CLI](#).

Você pode usar a CLI do CloudHSM e o par de chaves para criar um novo usuário administrador com o MFA habilitado.

Configurar o MFA para a CLI do CloudHSM

Siga estas etapas para configurar a autenticação multifator (MFA) para a CLI do CloudHSM.

1. Para configurar o MFA usando a Token Sign Strategy, você deve primeiro gerar uma chave privada RSA de 2048 bits e a chave pública associada.

```
$ openssl genrsa -out officer1.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
.....+++++
e is 65537 (0x010001)

$ openssl rsa -in officer1.key -outform PEM -pubout -out officer1.pub
writing RSA key
```

2. Digite o seguinte comando para iniciar o CLI no modo interativo.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

3. Usando a CLI do CloudHSM, faça login na sua conta de usuário.

```
aws-cloudhsm > login --username <admin> --role <admin> --cluster-id <cluster ID>
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "<admin>"
  }
}
```

4. Em seguida, execute o comando para alterar sua estratégia de MFA. Você deve fornecer o parâmetro `--token`. Esse parâmetro especifica um arquivo que terá tokens não assinados gravados nele.

```
aws-cloudhsm > user change-mfa token-sign --token unsigned-tokens.json --
username <username> --role crypto-user --change-quorum
Enter password:
Confirm password:
```

5. Agora você tem um arquivo com tokens não assinados que precisam ser assinados: `unsigned-tokens.json`. O número de tokens nesse arquivo depende do número de HSMs no seu cluster. Cada token representa um HSM. Esse arquivo está no formato JSON e contém tokens que precisam ser assinados para provar que você tem uma chave privada.

```
$ cat unsigned-tokens.json
{
  "version": "2.0",
  "tokens": [
    {
      {
```

```

    "unsigned": "Vtf/9Q0FY45v/E1osvpEMr59JsnP/hLDm4It002vqL8=",
    "signed": ""
  },
  {
    "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
    "signed": ""
  },
  {
    "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
    "signed": ""
  }
]
}

```

6. A próxima etapa é assinar esses tokens com a chave privada criada na etapa 1. Coloque as assinaturas de volta no arquivo. Primeiro, você precisa extrair e decodificar os tokens codificados em base64.

```

$ echo "Vtf/9Q0FY45v/E1osvpEMr59JsnP/hLDm4It002vqL8=" > token1.b64
$ echo "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=" > token2.b64
$ echo "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=" > token3.b64
$ base64 -d token1.b64 > token1.bin
$ base64 -d token2.b64 > token2.bin
$ base64 -d token3.b64 > token3.bin

```

7. Agora, você tem tokens binários que podem ser assinados usando a chave privada RSA criada na etapa 1.

```

$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token1.bin \
  -out token1.sig.bin
$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token2.bin \
  -out token2.sig.bin
$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \

```



```
-keyform PEM \
-in token3.bin \
-out token3.sig.bin
```

8. Agora você tem assinaturas binárias dos tokens. Codifique-os usando base64 e coloque-os novamente em seu arquivo de token.

```
$ base64 -w0 token1.sig.bin > token1.sig.b64
$ base64 -w0 token2.sig.bin > token2.sig.b64
$ base64 -w0 token3.sig.bin > token3.sig.b64
```

9. Finalmente, você pode copiar e colar os valores de base64 de volta em seu arquivo de token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwx9bJ0UUQLiNb7mxXS1uBJSExh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Q1q3W1Jh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2t1F9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGB0j4yB9recp0BB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASpvnNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+Hjmr0530QpZaJ/VXGIgApD0myuu/
ZGNKQTCskkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWr13JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAx0RTLlmwyU0YvPY0vUhc
+s83hx36QpGwGcD7RA0bPT50rTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93Sb0UyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnsduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD21FYppcgN1SzkY1ftr5E0jqS9ZjYEggGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96Qok1Nb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCi0YDzUCd43GftGq2LfxH3qSD51oFHg1HQV0Y0jyVzz1Avub5HQdt0Q
    }
  ]
}
```

10. Agora que seu arquivo de token tem todas as assinaturas necessárias, você pode continuar. Insira o nome do arquivo que contém os tokens assinados e pressione a tecla enter. Por fim, insira o caminho da sua chave pública.

```
Enter signed token file path (press enter if same as the unsigned token file):
Enter public key PEM file path:officer1.pub
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "crypto-user"
  }
}
```

Agora você configurou seu usuário com MFA.

```
{
  "username": "<username>",
  "role": "crypto-user",
  "locked": "false",
  "mfa": [
    {
      "strategy": "token-sign",
      "status": "enabled"
    }
  ],
  "cluster-coverage": "full"
},
```

Crie usuários com MFA habilitado para a CLI do CloudHSM

Siga estas etapas para criar AWS CloudHSM usuários com a autenticação multifator (MFA) ativada.

1. Use a CLI do CloudHSM para fazer login no HSM como administrador.
2. Use o [user create](#) comando para criar um usuário de sua escolha. Em seguida, siga as etapas [Configurar o MFA para a CLI do CloudHSM](#) para configurar o MFA para o usuário.

Faça login com os usuários com MFA habilitado para a CLI do CloudHSM

Siga estas etapas para fazer login de AWS CloudHSM usuários com a autenticação multifator (MFA) ativada.

1. Use o [login mfa-token-sign](#) comando na CLI do CloudHSM para iniciar o processo de login com o MFA para um usuário que tenha o MFA habilitado.

```
aws-cloudhsm > login --username <username> --role <role> mfa-token-sign --  
token <unsigned-tokens.json>  
Enter password:
```

2. Insira a senha. Em seguida, você será solicitado a inserir o caminho para o arquivo de token que contém unsigned/signed pares de tokens, onde tokens assinados são aqueles gerados usando sua chave privada.

```
aws-cloudhsm > login --username <username> --role <role> mfa-token-sign --  
token <unsigned-tokens.json>  
Enter password:  
Enter signed token file path (press enter if same as the unsigned token file):
```

3. Quando solicitado a inserir o caminho do arquivo de token assinado, você pode inspecionar o arquivo de token não assinado em um terminal separado. Identifique o arquivo com tokens não assinados que precisam ser assinados: *<unsigned-tokens.json>*. O número de tokens nesse arquivo depende do número de HSMs no seu cluster. Cada token representa um HSM. Esse arquivo está no formato JSON e contém tokens que precisam ser assinados para provar que você tem uma chave privada.

```
$ cat <unsigned-tokens.json>  
{  
  "version": "2.0",  
  "tokens": [  
    {  
      "unsigned": "Vtf/9Q0FY45v/E1osvpEMr59JsnP/hLDm4It002vqL8=",  
      "signed": ""  
    },  
    {  
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUd1cxLwZ4=",  
      "signed": ""  
    },  
    {  
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUd1cxLwZ4=",  
      "signed": ""  
    }  
  ]  
}
```

```

    "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
    "signed": ""
  }
]
}

```

- Assine os tokens não assinados com a chave privada criada na etapa 2. Primeiro, você precisa extrair e decodificar os tokens codificados em base64.

```

$ echo "VtF/9Q0FY45v/E1osvpEMr59JsnP/hLDm4It002vqL8=" > token1.b64
$ echo "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUd1cxLwZ4=" > token2.b64
$ echo "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=" > token3.b64
$ base64 -d token1.b64 > token1.bin
$ base64 -d token2.b64 > token2.bin
$ base64 -d token3.b64 > token3.bin

```

- Agora você tem tokens binários. Assine-os usando a chave privada RSA que você criou anteriormente na [etapa 1 da configuração da MFA](#).

```

$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token1.bin \
  -out token1.sig.bin
$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token2.bin \
  -out token2.sig.bin
$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token3.bin \
  -out token3.sig.bin

```

- Agora você tem assinaturas binárias dos tokens. Codifique-os usando base64 e coloque-os novamente em seu arquivo de token.

```

$ base64 -w0 token1.sig.bin > token1.sig.b64
$ base64 -w0 token2.sig.bin > token2.sig.b64

```

```
$ base64 -w0 token3.sig.bin > token3.sig.b64
```

7. Finalmente, você pode copiar e colar os valores de base64 de volta em seu arquivo de token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwx9bJ0UUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Q1q3W1Jh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2t1F9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGB0j4yB9recp0BB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASpNvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+Hjmr0530QpZaJ/VXGIgApD0myuu/
ZGNKQTCskkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRD4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAx0RTLlmwyU0YvPY0vUhc
+s83hx36QpGwGcD7RA0bPT50rTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93Sb0UyZRjH8=",
      "signed": "VgQPvrTsvG1jVBFxHnsduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgN1SzkYlfttr5E0jqS9ZjYEggGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96Qok1Nb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCi0YDzUCd43GftGq2LfxH3qSD51oFHg1HQV0Y0jyVzz1Avub5HQdt0Q
    }
  ]
}
```

8. Agora que seu arquivo de token tem todas as assinaturas necessárias, você pode continuar. Insira o nome do arquivo que contém os tokens assinados e pressione a tecla enter. Agora você deve fazer login com sucesso.

```
aws-cloudhsm > login --username <username> --role <role> mfa-token-sign --
token <unsigned-tokens.json>
Enter password:
Enter signed token file path (press enter if same as the unsigned token file):
```

```
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "<role>"
  }
}
```

Rotacione as chaves para usuários com MFA habilitado para CLI do CloudHSM

Siga estas etapas para alternar as chaves para AWS CloudHSM usuários com a autenticação multifator (MFA) ativada.

<result>

Você assinou o arquivo de token formatado em JSON gerado com sua chave privada e registrou uma nova chave pública de MFA.

</result>

1. Use a CLI do CloudHSM para fazer login no HSM como qualquer administrador ou como o usuário específico que tem o MFA ativado ([consulte Fazer login](#) de usuários com o MFA habilitado para obter detalhes).
2. Em seguida, execute o comando para alterar sua estratégia de MFA. Você deve fornecer o parâmetro `--token`. Esse parâmetro especifica um arquivo que terá tokens não assinados gravados nele.

```
aws-cloudhsm > user change-mfa token-sign --token unsigned-tokens.json --
username <username> --role crypto-user --change-quorum
Enter password:
Confirm password:
```

3. Identifique o arquivo com tokens não assinados que precisam ser assinados: `unsigned-tokens.json`. O número de tokens nesse arquivo depende do número de HSMs no seu cluster. Cada token representa um HSM. Esse arquivo está no formato JSON e contém tokens que precisam ser assinados para provar que você tem uma chave privada. Essa será a nova chave privada do novo par de public/private chaves RSA que você deseja usar para rotacionar a chave pública atualmente registrada.

```
$ cat unsigned-tokens.json
```

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "Vtf/9Q0FY45v/E1osvpEMr59JsnP/hLDm4It002vqL8=",
      "signed": ""
    },
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUd1cxLwZ4=",
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h81hTVt9oLbxppjod0Ebysydw=",
      "signed": ""
    }
  ]
}
```

4. Assine esses tokens com a chave privada que você criou anteriormente durante a configuração. Primeiro, precisamos extrair e decodificar os tokens codificados em base64.

```
$ echo "Vtf/9Q0FY45v/E1osvpEMr59JsnP/hLDm4It002vqL8=" > token1.b64
$ echo "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUd1cxLwZ4=" > token2.b64
$ echo "z6aW9RzErJBL5KqFG5h81hTVt9oLbxppjod0Ebysydw=" > token3.b64
$ base64 -d token1.b64 > token1.bin
$ base64 -d token2.b64 > token2.bin
$ base64 -d token3.b64 > token3.bin
```

5. Agora você tem tokens binários. Assine-os usando a chave privada RSA que você criou anteriormente durante a configuração.

```
$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token1.bin \
  -out token1.sig.bin
$ openssl pkeyutl -sign \
  -inkey officer1.key \
  -pkeyopt digest:sha256 \
  -keyform PEM \
  -in token2.bin \
  -out token2.sig.bin
```

```
$ openssl pkeyutl -sign \
    -inkey officer1.key \
    -pkeyopt digest:sha256 \
    -keyform PEM \
    -in token3.bin \
    -out token3.sig.bin
```

6. Agora você tem assinaturas binárias dos tokens. Codifique-os usando base64 e coloque-os novamente em seu arquivo de token.

```
$ base64 -w0 token1.sig.bin > token1.sig.b64
$ base64 -w0 token2.sig.bin > token2.sig.b64
$ base64 -w0 token3.sig.bin > token3.sig.b64
```

7. Finalmente, você pode copiar e colar os valores de base64 de volta em seu arquivo de token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwx9bJ0UUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Q1q3W1Jh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2t1F9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGB0j4yB9recp0BB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASpvnNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+Hjmr0530QpZaJ/VXGIGApD0myuu/
ZGNKQTCskkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWr13JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAx0RTLlmwyU0YvPY0vUhc
+s83hx36QpGwGcD7RA0bPT50rTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93Sb0UyZRjH8=",
      "signed": "VgQPvrTsvG1jVBFxHnsduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD21FYppcgN1SzkY1ftr5E0jqS9ZjYEggGuB4g//
MxaBaRbJai/6B1cE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96Qok1Nb1WUuSHw
+psUyeIVtIwFMHEffoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCi0YDzUCd43GftGq2LfxH3qSD51oFHg1HQV0Y0jyVzz1Avub5HQdt0Q
```



```

    }
  ]
}

```

8. Agora que seu arquivo de token tem todas as assinaturas necessárias, você pode continuar. Insira o nome do arquivo que contém os tokens assinados e pressione a tecla enter. Por fim, insira o caminho da sua nova chave pública. Agora você verá o seguinte como parte da saída da [lista de usuários](#).

```

Enter signed token file path (press enter if same as the unsigned token file):
Enter public key PEM file path:officer1.pub
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "crypto-user"
  }
}

```

Agora configuramos nosso usuário com MFA.

```

{
  "username": "<username>",
  "role": "crypto-user",
  "locked": "false",
  "mfa": [
    {
      "strategy": "token-sign",
      "status": "enabled"
    }
  ],
  "cluster-coverage": "full"
},

```

Cancelar o registro de uma chave pública de MFA usando a CLI do CloudHSM

Siga estas etapas para cancelar o registro de uma chave pública de autenticação multifator (MFA) para usuários administradores do AWS CloudHSM quando a chave pública de MFA for registrada.

1. Use a CLI do CloudHSM para fazer login no HSM como administrador com MFA ativado.

2. Use o `user change-mfa token-sign` comando para remover o MFA de um usuário.

```
aws-cloudhsm > user change-mfa token-sign --username <username> --role admin --
deregister --change-quorum
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "admin"
  }
}
```

Referência de arquivo de token para MFA com CLI do CloudHSM

O arquivo de token gerado ao registrar uma chave pública de autenticação multifator (MFA) ou ao tentar fazer login na CLI do CloudHSM usando a MFA consiste no seguinte:

- Tokens: uma matriz de pares de unsigned/signed tokens codificados em base64 na forma de literais de objetos JSON.
- Não assinado: um token codificado e SHA256 com hash em base64.
- Assinado: um token assinado codificado em base64 (assinatura) do token não assinado, usando a chave privada RSA de 2048 bits.

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwx9bJ0UUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Q1q3W1Jh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2t1F9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/TK0PVaxLN42X
+pebV4juwMhN4mK4CzdFAJgM+UGB0j4yB9recp0BB9K8QFSpJZALSEdDgUc/mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37+j/
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASpvnNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
```

```

    "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+Hjmr0530QpZaJ/VXGIgApD0myuu/
ZGNKQTCskkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAx0RTLlmwyU0YvPY0vUhc
+s83hx36QpGwGcD7RA0bPT50rTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
  },
  {
    "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93Sb0UyZRjH8=",
    "signed": "VgQPvrTsvGljVBFxHnsduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlfttr5E0jqS9ZjYEggGuB4g//
MxaBaRbJai/6BlcE92NidBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96Qok1Nb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCi0YDzUCd43GftGq2LfxH3qSD51oFHg1HQV0Y0jyVzzlAvub5HQdt0QdErI
  }
]
}

```

Gerenciar a autenticação de quórum (controle de acesso M ou N) usando a CLI do CloudHSM

AWS CloudHSM os clusters oferecem suporte à autenticação de quorum, também conhecida como controle de acesso M of N. Esse atributo exige que os usuários do HSM cooperem em determinadas operações, adicionando uma camada extra de proteção.

Com a autenticação de quórum, nenhum usuário único no HSM pode realizar operações controladas pelo quórum no HSM. Em vez disso, um número mínimo de usuários do HSM (pelo menos 2) deve cooperar para realizar essas operações.

A autenticação de quorum pode controlar as seguintes operações:

- Gerenciamento de usuários do HSM por [administrador](#): criar e excluir usuários do HSM ou alterar a senha de um usuário do HSM diferente. Para obter mais informações, consulte [Gerenciamento de usuários com autenticação de quorum habilitada para usar o CloudHSM AWS CloudHSM CLI](#).

Pontos-chave sobre a autenticação de quórum no AWS CloudHSM.

- Um usuário do HSM pode assinar seu próprio token de quórum, ou seja, fornecer uma das aprovações necessárias para a autenticação de quórum.
- Você escolhe o número mínimo de aprovadores de quórum, que varia de dois (2) a oito (8).

- HSMs pode armazenar até 1024 tokens de quórum. Quando esse limite é atingido, o HSM limpa um token expirado para criar um novo.
- Por padrão, os tokens expiram dez minutos após sua criação.
- Se os clusters com MFA estiverem ativados, será usada a mesma chave para autenticação de quórum e para autenticação multifator (MFA). Consulte [Usar a CloudHSM CLI para gerenciar a MFA](#) para obter mais informações.
- Cada HSM pode conter apenas um token por serviço administrativo por vez, mas vários tokens por serviço de Usuário de criptografia.

Os seguintes tópicos fornecem mais informações sobre a autenticação de quorum no AWS CloudHSM.

Tópicos

- [Processo de autenticação de quórum para a CLI do CloudHSM](#)
- [Nomes e tipos AWS CloudHSM de serviços compatíveis para autenticação de quórum com o CloudHSM CLI](#)
- [Configurar a autenticação de quórum para AWS CloudHSM administradores usando o CloudHSM CLI](#)
- [Gerenciamento de usuários com autenticação de quorum habilitada para usar o CloudHSM AWS CloudHSM CLI](#)
- [Alterar o valor mínimo do quórum para usar a CLI do AWS CloudHSM CloudHSM](#)

Processo de autenticação de quórum para a CLI do CloudHSM

As etapas a seguir resumem os processos de autenticação de quórum para a CLI do CloudHSM. Para conhecer as etapas e ferramentas específicas, consulte [Gerenciamento de usuários com autenticação de quorum habilitada para usar o CloudHSM AWS CloudHSM CLI](#).

1. Cada usuário do módulo de segurança de hardware (HSM) cria uma chave assimétrica para assinatura. Isso é feito fora do HSM, tomando cuidado para proteger a chave apropriadamente.
2. Cada usuário do HSM faz login no HSM e registra a parte pública de sua chave de assinatura (a chave pública) no HSM.
3. Quando um usuário do HSM quer fazer uma operação controlada por quorum, ele faz login no HSM e obtém um token de quorum.

4. O usuário do HSM fornece esse token de quorum a um ou mais usuários do HSM e solicita sua aprovação.
5. Os outros usuários do HSM aprovam usando suas chaves para assinar criptograficamente o token de quorum. Isso ocorre fora do HSM.
6. Quando o usuário do HSM tem o número necessário de aprovações, ele faz login no HSM e executa a operação controlada por quórum com o argumento `--approval`, fornecendo o arquivo de token de quórum assinado, que contém todas as aprovações (assinaturas) necessárias.
7. O HSM usa as chaves públicas registradas de cada assinante para verificar as assinaturas. Se as assinaturas forem válidas, o HSM aprova o token e a operação controlada por quórum será executada.

Nomes e tipos AWS CloudHSM de serviços compatíveis para autenticação de quórum com o CloudHSM CLI

Serviços administrativos: a autenticação de quórum é usada para serviços com privilégios administrativos, como criar usuários, excluir usuários, alterar senhas de usuários, definir valores de quórum e desativar recursos de quórum e MFA.

Serviços de criptografia para usuários: a autenticação de quórum é usada para serviços privilegiados de usuários criptográficos associados a uma chave específica, como assinar com uma chave, uma chave, `sharing/unsharing` uma chave e definir o atributo de `wrapping/unwrapping` uma chave. O valor do quórum de uma chave associada é configurado quando a chave é gerada, importada ou desencapsulada. O valor do quórum deve ser igual ou menor do que o número de usuários aos quais a chave está associada, o que inclui usuários com os quais a chave é compartilhada e o proprietário da chave.

Cada tipo de serviço é subdividido em um nome de serviço qualificado, que contém um conjunto específico de operações de serviço suportadas por quórum que podem ser executadas.

Nome do serviço	Tipo de serviço	Operações de serviço
usuário	Administrador	- criar usuário - excluir usuário - alterar senha de usuário - user change-mfa

Nome do serviço	Tipo de serviço	Operações de serviço
quorum	Administrador	• sinal simbólico de quórum set-quorum-value
cluster ¹	Administrador	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gerenciamento de chaves	Usuário de criptografia	• encapsulamento de chaves • desencapsulamento de chaves • compartilhamento de chaves • descompartilhamento de chaves • key set-attribute
uso de chave	Usuário de criptografia	• sinal de chave

[1] O serviço de cluster está disponível exclusivamente em instâncias hsm2m.medium

Configurar a autenticação de quórum para AWS CloudHSM administradores usando o CloudHSM CLI

Os tópicos a seguir descrevem as etapas que você deve concluir para configurar seu módulo de segurança de hardware (HSM) para que os AWS CloudHSM [administradores](#) possam usar a autenticação de quorum. Você precisa realizar essas etapas apenas uma vez ao configurar inicialmente a autenticação de quorum para administradores. Depois de concluir essas etapas, consulte [Gerenciamento de usuários com autenticação de quorum habilitada para usar o CloudHSM AWS CloudHSM CLI](#).

Tópicos

- [Pré-requisitos](#)

- [Etapa 1. Criar e registrar uma chave para assinatura](#)
- [Etapa 2. Definir o valor mínimo do quorum no HSM](#)
- [Valores mínimos de quórum](#)

Pré-requisitos

Para entender esse exemplo, você deve estar familiarizado com o [CloudHSM CLI](#).

Etapa 1. Criar e registrar uma chave para assinatura

Para usar a autenticação de quórum, cada administrador deve concluir todas as etapas a seguir:

Tópicos

- [Criar um par de chaves RSA](#)
- [Crie e assine um token de registro](#)
- [Registrar uma chave pública no HSM](#)

Criar um par de chaves RSA

Existem muitas maneiras diferentes de criar e proteger um par de chaves. O exemplo a seguir mostra como fazer isso com o [OpenSSL](#).

Exemplo Crie uma chave privada com o OpenSSL

O exemplo a seguir demonstra como usar o OpenSSL para criar uma chave RSA de 2048 bits. Para usar esse exemplo, *<admin.key>* substitua pelo nome do arquivo em que você deseja armazenar a chave.

```
$ openssl genrsa -out <admin.key>
Generating RSA private key, 2048 bit long modulus
.....+++
.+++
e is 65537 (0x10001)
```

Em seguida, gere a chave pública usando a chave privada que você acabou de criar.

Exemplo Crie uma chave pública com o OpenSSL

O exemplo a seguir demonstra como usar o OpenSSL para criar uma chave pública a partir da chave privada que você acabou de criar.

```
$ openssl rsa -in admin.key -outform PEM -pubout -out admin1.pub  
writing RSA key
```

Crie e assine um token de registro

Você cria um token e o assina com a chave privada que acabou de gerar na etapa anterior.

Example Crie um token de registro

1. Use o comando a seguir para iniciar CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Crie um token de registro executando o comando [quorum token-sign generate](#):

```
aws-cloudhsm > quorum token-sign generate --service registration --token /path/  
tokenfile  
{  
  "error_code": 0,  
  "data": {  
    "path": "/path/tokenfile"  
  }  
}
```

3. O comando [quorum token-sign generate](#) gera um token de registro no caminho de arquivo especificado. Inspecione o arquivo de token:

```
$ cat /path/tokenfile  
{  
  "version": "2.0",  
  "tokens": [  
    {  
      "approval_data": <approval data in base64 encoding>,  
      "unsigned": <unsigned token in base64 encoding>,  
      "signed": ""  
    }  
  ]  
}
```



```
]
}
```

O caminho do arquivo consiste no seguinte:

- `approval_data`: um token de dados aleatório codificado em base64 cujos dados brutos não excedem o máximo de 245 bytes.
- `unsigned`: um token codificado em base64 e com SHA256 hash do `approval_data`.
- `signed`: um token assinado codificado em base64 (assinatura) do token não assinado, usando a chave privada RSA de 2048 bits gerada anteriormente com o OpenSSL.

Você assina o token não assinado com a chave privada para demonstrar que tem acesso à chave privada. Você precisará do arquivo de token de registro totalmente preenchido com uma assinatura e a chave pública para registrar o administrador como usuário de quórum no cluster.

AWS CloudHSM

Example Assine o token de registro não assinado

1. Decodifique o token não assinado codificado em base64 e coloque-o em um arquivo binário:

```
$ echo -n '6BMUj6mUjjko6ZLCEdzG1WpR5sILhFJfqhW1ej30q1g=' | base64 -d > admin.bin
```

2. Use OpenSSL e a chave privada para assinar o token de registro não assinado, agora binário, e criar um arquivo de assinatura binária:

```
$ openssl pkeyutl -sign \
-inkey admin.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in admin.bin \
-out admin.sig.bin
```

3. Codifique a assinatura binária em base64:

```
$ base64 -w0 admin.sig.bin > admin.sig.b64
```

4. Copie e cole a assinatura codificada em base64 no arquivo de token:

```
{
```

```
"version": "2.0",
"tokens": [
  {
    "approval_data": <approval data in base64 encoding>,
    "unsigned": <unsigned token in base64 encoding>,
    "signed": <signed token in base64 encoding>
  }
]
}
```

Registrar uma chave pública no HSM

Depois de criar uma chave, o administrador deve registrar a chave pública no AWS CloudHSM cluster.

Para registrar uma chave pública no HSM

1. Use o comando a seguir para iniciar CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CLI do CloudHSM, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Use o comando [Registrar a estratégia de quórum de assinatura de token de um usuário usando a CLI do CloudHSM](#) para registrar a chave pública. Para obter mais informações, consulte o exemplo a seguir ou use o comando `help user change-quorum token-sign register`.

Example— Registrar uma chave pública com o AWS CloudHSM cluster

O exemplo a seguir mostra como usar o comando `user change-quorum token-sign register` na CloudHSM CLI para registrar a chave pública de um administrador no HSM. Para usar esse comando, o CO deve estar conectado no HSM. Substitua estes valores pelos seus próprios:

```
aws-cloudhsm > user change-quorum token-sign register --public-key </path/admin.pub> --signed-token </path/tokenfile>
{
  "error_code": 0,
  "data": {
    "username": "admin",
    "role": "admin"
  }
}
```

 Note

/path/admin.pub: o caminho do arquivo PEM de chave pública

Obrigatório: sim

/path/tokenfile: o caminho do arquivo com token assinado pela chave privada do usuário

Obrigatório: sim

Depois que todos os administradores registrarem suas chaves públicas, a saída do comando `user list` mostra essa informação no campo de quórum, informando a estratégia de quórum ativada em uso, conforme mostrado abaixo:

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
```

```
"locked": "false",
"mfa": [],
"quorum": [
  {
    "strategy": "token-sign",
    "status": "enabled"
  }
],
"cluster-coverage": "full"
},
{
  "username": "admin2",
  "role": "admin",
  "locked": "false",
  "mfa": [],
  "quorum": [
    {
      "strategy": "token-sign",
      "status": "enabled"
    }
  ],
  "cluster-coverage": "full"
},
{
  "username": "admin3",
  "role": "admin",
  "locked": "false",
  "mfa": [],
  "quorum": [
    {
      "strategy": "token-sign",
      "status": "enabled"
    }
  ],
  "cluster-coverage": "full"
},
{
  "username": "admin4",
  "role": "admin",
  "locked": "false",
  "mfa": [],
  "quorum": [
    {
      "strategy": "token-sign",
```

```
        "status": "enabled"
      }
    ],
    "cluster-coverage": "full"
  },
  {
    "username": "app_user",
    "role": "internal(APPLIANCE_USER)",
    "locked": "false",
    "mfa": [],
    "quorum": [],
    "cluster-coverage": "full"
  }
]
}
```

Neste exemplo, o AWS CloudHSM cluster tem dois HSMs, cada um com os mesmos administradores, conforme mostrado na saída a seguir do user list comando. Para obter mais informações sobre a criação de usuários, consulte [Gerenciamento de usuários com a CLI do CloudHSM](#)

Etapas 2. Definir o valor mínimo do quorum no HSM

Para usar a autenticação de quórum, um administrador deve fazer login no HSM e, em seguida, definir o valor mínimo de quorum. Esse é o número mínimo de aprovações de administrador necessárias para realizar operações de gerenciamento de usuários do HSM. Qualquer administrador no HSM pode definir o valor mínimo de quorum, incluindo administradores que não tenham registrado uma chave para assinatura. Você pode alterar o valor mínimo de quórum a qualquer momento. Para obter mais informações, consulte [Altere o valor mínimo](#).

Para definir o valor mínimo do quorum no HSM

1. Use o comando a seguir para iniciar CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CLI do CloudHSM, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Use o comando [Atualizar um valor de quórum usando a CLI do CloudHSM](#) para definir o valor mínimo do quorum. O sinalizador `--service` identifica o serviço HSM para o qual você está definindo valores. Consulte o exemplo a seguir ou use o comando `help quorum token-sign set-quorum-value` para obter mais informações.

Example Definir o valor mínimo do quorum no HSM

Este exemplo usa um valor mínimo de quorum de dois (2). Você pode escolher qualquer valor de dois (2) até oito (8), até o número total de administradores no HSM. Neste exemplo, o HSM tem quatro (4) administradores, então o valor máximo possível é quatro (4).

Para usar o comando de exemplo a seguir, substitua o número final (`<2>`) pelo valor mínimo de quórum preferido.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service user --value <2>
{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

Neste exemplo, o comando [Mostrar valores de quórum usando a CLI do CloudHSM](#) lista os tipos, os nomes e as descrições de serviço do HSM que estão incluídos no serviço.

Valores mínimos de quórum

Use o comando `quorum token-sign list-quorum-values` para obter o valor mínimo de quorum para um serviço:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 1
  }
}
```

A saída do comando `quorum token-sign list-quorum-values` anterior mostra que o valor mínimo de quorum para as operações de gerenciamento de usuários do HSM, responsável pelas operações de gerenciamento de usuário, é agora dois (2). Depois de concluir essas etapas, consulte [Gerenciamento de usuários com quórum \(M de N\)](#).

Serviços administrativos: a autenticação de quórum é usada para serviços com privilégios administrativos, como criar usuários, excluir usuários, alterar senhas de usuários, definir valores de quórum e desativar recursos de quórum e MFA.

Serviços de criptografia para usuários: a autenticação de quórum é usada para serviços privilegiados de usuários criptográficos associados a uma chave específica, como assinar com uma chave, uma chave, `sharing/unsharing` uma chave e definir o atributo de `wrapping/unwrapping` uma chave. O valor do quórum de uma chave associada é configurado quando a chave é gerada, importada ou desencapsulada. O valor do quórum deve ser igual ou menor do que o número de usuários aos quais a chave está associada, o que inclui usuários com os quais a chave é compartilhada e o proprietário da chave.

Cada tipo de serviço é subdividido em um nome de serviço qualificado, que contém um conjunto específico de operações de serviço suportadas por quórum que podem ser executadas.

Nome do serviço	Tipo de serviço	Operações de serviço
usuário	Administrador	- criar usuário - excluir usuário - alterar senha de usuário

Nome do serviço	Tipo de serviço	Operações de serviço
		• user change-mfa
quorum	Administrador	• sinal simbólico de quórum • set-quorum-value
cluster ¹	Administrador	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gerenciamento de chaves	Usuário de criptografia	• encapsulamento de chaves • desencapsulamento de chaves • compartilhamento de chaves • descompartilhamento de chaves • key set-attribute
uso de chave	Usuário de criptografia	• sinal de chave

[1] O serviço de cluster está disponível exclusivamente em instâncias hsm2m.medium

Gerenciamento de usuários com autenticação de quorum habilitada para usar o CloudHSM AWS CloudHSM CLI

Um AWS CloudHSM [administrador](#) no módulo de segurança de hardware (HSM) pode configurar a autenticação de quorum para as seguintes operações no cluster: AWS CloudHSM

- [Crie um AWS CloudHSM usuário com o CloudHSM CLI](#)
- [Excluir um AWS CloudHSM usuário com a CLI do CloudHSM](#)
- [Alterar a senha de um usuário com a CLI do CloudHSM](#)
- [A categoria user change-mfa na CLI do CloudHSM](#)

Depois que o AWS CloudHSM cluster é configurado para autenticação de quórum, os administradores não podem realizar as operações de gerenciamento de usuários do HSM sozinhos. O exemplo a seguir mostra a saída quando um administrador tenta criar um novo usuário no HSM. O comando falha com um erro, informando que a autenticação de quórum é necessária.

```
aws-cloudhsm > user create --username user1 --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 1,
  "data": "Quorum approval is required for this operation"
}
```

Para realizar uma operação de gerenciamento de usuário do HSM, um administrador deve completar as seguintes tarefas:

Tópicos

- [Etapa 1. Obter um token de quorum](#)
- [Etapa 2. Obter assinaturas dos administradores de aprovação](#)
- [Etapa 3. Aprovar o token no AWS CloudHSM cluster e executar uma operação de gerenciamento de usuários](#)

Etapa 1. Obter um token de quorum

Primeiro, o administrador deve usar a CLI do CloudHSM para solicitar um token de quórum.

Para obter um token de quorum

1. Use o comando a seguir para iniciar CloudHSM CLI.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CloudHSM CLI, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Use o comando `quorum token-sign generate` para obter um token de quorum. Para obter mais informações, consulte o exemplo a seguir ou use o comando `help quorum token-sign generate`.

Example Gere um token de quórum

Este exemplo obtém um token de quorum para o administrador com o nome de usuário `admin` e salva o token em um arquivo chamado `admin.token`. Para usar o exemplo de comando, substitua esses valores por seus próprios:

- **<admin>**— O nome do administrador que está recebendo o token. Este deve ser o mesmo administrador que está conectado ao HSM e executando esse comando.
- **<admin.token>**— O nome do arquivo a ser usado para armazenar o token de quorum.

No comando a seguir, `user` identifica o nome do serviço para o qual você pode usar o token que você está gerando. Nesse caso, o token é para operações de gerenciamento de usuários do HSM (serviço `user`).

```
aws-cloudhsm > login --username <admin> --role admin --password <password>
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

```
aws-cloudhsm > quorum token-sign generate --service user --token </path/admin.token>
{
  "error_code": 0,
```

```
"data": {
  "path": "/home/tfile"
}
```

O comando `quorum token-sign generate` gera um token de quórum de serviço de usuário no caminho de arquivo especificado. O arquivo de token pode ser inspecionado:

```
$ cat </path/admin.token>
{
  "version": "2.0",
  "service": "user-management",
  "approval_data": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJA1fK
+0WbNhZG1pbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABj5vbeAAAAAAAAAAAAAQADAAAFQAAAAAAAAAAW/
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAAABAAMAAAAUA
+b23gAAAAAAAA",
  "token": "012LZkmAHZyAc1hPhyck0oVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": []
}
```

O caminho do arquivo consiste no seguinte:

- `service`: um identificador do serviço de quórum ao qual o token está associado.
- `approval_data`: um token de dados brutos codificado em base64 gerado pelo HSM.
- `token`: um token codificado em base64 e com hash SHA-256 do `approval_data`
- `signatures`: uma matriz de tokens assinados codificados em base64 (assinaturas) do token não assinado, em que cada assinatura de um aprovador está no formato literal de objeto JSON:

```
{
  "username": "<APPROVER_USERNAME>",
  "role": "<APPROVER_ROLE>",
  "signature": "<APPROVER_RSA2048_BIT_SIGNATURE>"
}
```

Cada assinatura é criada do resultado de um aprovador usando sua chave privada RSA de 2048 bits correspondente, cuja chave pública foi registrada no HSM.

É possível confirmar a existência do token de quórum do serviço de usuário gerado no cluster do CloudHSM executando o comando `quorum token-sign list`:

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": [
      {
        "username": "admin",
        "service": "user",
        "approvals-required": {
          "value": 2
        },
        "number-of-approvals": {
          "value": 0
        },
        "token-timeout-seconds": {
          "value": 597
        },
        "cluster-coverage": "full"
      }
    ]
  }
}
```

O tempo `token-timeout-seconds` indica o período de tempo limite em segundos para que um token gerado seja aprovado antes de expirar.

Etapa 2. Obter assinaturas dos administradores de aprovação

Um administrador que tenha um token de quorum deve fazer com que o token seja aprovado por outros administradores. Para darem sua aprovação, os outros administradores usam suas chaves de assinatura para assinar criptograficamente o token. Eles fazem isso fora do HSM.

Existem muitas maneiras diferentes de assinar o token. O exemplo a seguir mostra como fazer isso com o [OpenSSL](#). Para usar uma ferramenta de assinatura diferente, certifique-se de que essa ferramenta use a chave privada (chave de assinatura) do administrador para assinar um resumo SHA-256 do token.

Exemplo Obter assinaturas dos administradores de aprovação

Neste exemplo, o administrador que possui o token (`admin`) precisa de pelo menos duas (2) aprovações. Os comandos do exemplo a seguir mostram como dois (2) administradores podem usar o OpenSSL para assinar criptograficamente o token.

1. Decodifique o token não assinado codificado em base64 e coloque-o em um arquivo binário:

```
$ echo -n '012LZkmAHZyAc1hPhyck0oVW33aGrgG77qmDHWQ3CJ8=' | base64 -d > admin.bin
```

2. Use o OpenSSL e a respectiva chave privada do aprovador (admin3) para assinar o token não assinado de quórum, agora binário, para o serviço ao usuário e criar um arquivo de assinatura binária:

```
$ openssl pkeyutl -sign \
-inkey admin3.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in admin.bin \
-out admin.sig.bin
```

3. Codifique a assinatura binária em base64:

```
$ base64 -w0 admin.sig.bin > admin.sig.b64
```

4. Por fim, copie e cole a assinatura codificada em base64 no arquivo de token, seguindo o formato literal do objeto JSON especificado anteriormente para a assinatura do aprovador:

```
{
  "version": "2.0",
  "approval_data": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJA1fK
+0WbNhZG1pbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABj5vbeAAAAAAAAAAAAAQADAAAFQAAAAAAAAAAW
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAAABAAMAA
+b23gAAAAAAAA",
  "token": "012LZkmAHZyAc1hPhyck0oVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": [
    {
      "username": "admin2",
      "role": "admin",
      "signature": "06qx7/mUaVkyYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+0sDtvYujhEU4a0FVLcrUFmyB/CX90QmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1n0XwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWwdtDgft7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyrugFLpXs1n0tJ0EglGhx2qbYTs+omKWZd0R15WIWEXW3IXw/
Dg5vV0brNpvG0eZK08nSMc27+cyPySc+ZbNw=="
    },
    {
      "username": "admin3",
```

```

    "role": "admin",
    "signature": "06qx7/mUaVkyYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+0sDtvYujhEU4a0FVLcrUFmyB/CX90QmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1n0XwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWwdtDgftT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyruGFLpXs1n0tJ0EglGhx2qbYTs+omKWZd0R15WIWEXW3IXw/
Dg5vV0brNpvG0eZK08nSMc27+cyPySc+ZbNw=="
  }
]
}

```

Etapa 3. Aprovar o token no AWS CloudHSM cluster e executar uma operação de gerenciamento de usuários

Depois que um administrador tiver as aprovações/assinaturas necessárias, conforme detalhado na seção anterior, ele poderá fornecer esse token ao cluster do AWS CloudHSM junto com uma das seguintes operações de gerenciamento de usuários:

- [criar](#)
- [excluir](#)
- [change-password](#)
- [user change-mfa](#)

Para obter mais informações sobre como usar esses comandos, consulte [Gerenciamento de usuários com a CLI do CloudHSM](#).

Durante a transação, o token será aprovado no AWS CloudHSM cluster e executará a operação de gerenciamento de usuários solicitada. O sucesso da operação de gerenciamento de usuários depende tanto de um token de quórum válido aprovado quanto de uma operação válida de gerenciamento de usuários.

O administrador pode usar o token para apenas uma operação. Quando essa operação for bem-sucedida, o token não será mais válido. Para realizar outra operação de gerenciamento de usuários do HSM, o administrador deve repetir o processo descrito acima. Ou seja, o administrador deve obter um novo token de quorum, obter novas assinaturas dos aprovadores e, então, aprovar e consumir o novo token no HSM com a operação de gerenciamento de usuário solicitada.

 Note

O token de quórum só é válido enquanto sua sessão de login atual estiver aberta. Se você fizer log out da CLI do CloudHSM ou se a rede se desconectar, o token não será mais válido. Da mesma forma, um token autorizado só pode ser usado na CLI do CloudHSM. Ele não pode ser usado para autenticação em um aplicativo diferente.

Example Criar um novo usuário como administrador

No comando do exemplo a seguir, um administrador logado cria um novo usuário no HSM:

```
aws-cloudhsm > user create --username user1 --role crypto-user --approval /path/  
admin.token  
Enter password:  
Confirm password:  
{  
  "error_code": 0,  
  "data": {  
    "username": "user1",  
    "role": "crypto-user"  
  }  
}
```

O administrador, então, insere o comando `user list` para confirmar a criação do novo usuário:

```
aws-cloudhsm > user list  
{  
  "error_code": 0,  
  "data": {  
    "users": [  
      {  
        "username": "admin",  
        "role": "admin",  
        "locked": "false",  
        "mfa": [],  
        "quorum": [  
          {  
            "strategy": "token-sign",  
            "status": "enabled"  
          }  
        ],  
      },  
    ],  
  },  
}
```

```
    "cluster-coverage": "full"
  },
  {
    "username": "admin2",
    "role": "admin",
    "locked": "false",
    "mfa": [],
    "quorum": [
      {
        "strategy": "token-sign",
        "status": "enabled"
      }
    ],
    "cluster-coverage": "full"
  },
  {
    "username": "admin3",
    "role": "admin",
    "locked": "false",
    "mfa": [],
    "quorum": [
      {
        "strategy": "token-sign",
        "status": "enabled"
      }
    ],
    "cluster-coverage": "full"
  },
  {
    "username": "admin4",
    "role": "admin",
    "locked": "false",
    "mfa": [],
    "quorum": [
      {
        "strategy": "token-sign",
        "status": "enabled"
      }
    ],
    "cluster-coverage": "full"
  },
  {
    "username": "user1",
    "role": "crypto-user",
```



```

    "locked": "false",
    "mfa": [],
    "quorum": [],
    "cluster-coverage": "full"
  },
  {
    "username": "app_user",
    "role": "internal(APPLIANCE_USER)",
    "locked": "false",
    "mfa": [],
    "quorum": [],
    "cluster-coverage": "full"
  }
]
}
}

```

Se o administrador tentar executar outra operação de gerenciamento de usuário HSM, ele falhará com um erro de autenticação de quorum:

```

aws-cloudhsm > user delete --username user1 --role crypto-user
{
  "error_code": 1,
  "data": "Quorum approval is required for this operation"
}

```

Conforme mostrado abaixo, o comando `quorum token-sign list` mostra que o administrador não tem tokens aprovados. Para realizar outra operação de gerenciamento de usuários do HSM, o administrador deve gerar um novo token de quórum, obter novas assinaturas dos aprovadores e executar a operação de gerenciamento de usuário desejada com o argumento `--approval` para fornecer o token de quórum a ser aprovado e consumido durante a execução da operação de gerenciamento de usuários.

```

aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": []
  }
}

```

Alterar o valor mínimo do quórum para usar a CLI do AWS CloudHSM CloudHSM

Depois de [definir o valor mínimo do quórum](#) para [administradores](#) do CloudHSM, talvez seja necessário ajustar o valor mínimo do quórum. O HSM permite que você altere o valor mínimo de quórum somente quando o número de aprovadores é igual ou superior ao valor atual. Por exemplo, com um valor mínimo de dois (2) COs, pelo menos 2 (dois) COs deverão aprovar qualquer alteração.

Note

O valor do quórum do serviço ao usuário deve sempre ser menor ou igual ao valor do quórum do serviço de quórum. Para obter mais informações sobre nomes de serviço, consulte [Nomes e tipos AWS CloudHSM de serviços compatíveis para autenticação de quórum com o CloudHSM CLI](#).

Para obter a aprovação do quorum para alterar o valor mínimo do quorum, você precisa de um token de quorum para quorum service usando o comando quorum token-sign set-quorum-value. Para gerar um token de quórum para o quorum service usando o comando quorum token-sign set-quorum-value, o serviço de quórum deve ser maior que um (1). Isso significa que, antes que você possa alterar o valor mínimo de quorum para o serviço ao usuário, talvez seja necessário alterar o valor mínimo do serviço ao quorum.

Etapas para alterar o valor mínimo do quórum para administradores

1. Inicie o modo interativo da CloudHSM CLI.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CloudHSM CLI, faça login como administrador.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
```

```
"data": {
  "username": "<admin>",
  "role": "admin"
}
```

3. Verifique os valores mínimos do quórum atual:

```
aws-cloudhsm > quorum token-sign list-quorum-values
```

4. Se o valor mínimo do quórum para o serviço de quórum for menor do que o valor para o serviço ao usuário, altere o valor do serviço de quórum:

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value <3>
```

5. [Gere um token de quórum](#) para o serviço de quórum.
6. [Obtenha aprovações \(assinaturas\) de outros admins.](#)
7. [Aprove o token no cluster do CloudHSM e execute uma operação de gerenciamento de usuários.](#)
8. Altere o valor mínimo do quórum para o serviço de usuário:

```
aws-cloudhsm > quorum token-sign set-quorum-value
```

Example Ajustar os valores mínimos do serviço de quórum

1. Verifique os valores atuais. O comando de exemplo a seguir mostra que o valor mínimo do quórum para o serviço de usuário atualmente é dois (2).

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 1
  }
}
```

2. Altere o valor do serviço de quórum. Defina o valor mínimo do quórum para o serviço de quórum como igual ou maior do que o valor do serviço de usuário. Este exemplo define o valor mínimo do

quórum para o serviço de quórum como dois (2), o mesmo valor definido para o serviço de usuário no exemplo anterior.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value 2
{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

3. Verifique as alterações. Este comando mostra que o valor mínimo do quórum é agora dois (2) para o serviço de usuário e o serviço de quórum.

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 2
  }
}
```

Gerenciamento de usuários do HSM com o CloudHSM Management Utility (CMU)

Para gerenciar os usuários do módulo de segurança de hardware (HSM) AWS CloudHSM, você deve fazer login no HSM com o nome de usuário e a senha de um [oficial criptográfico](#) (CO). Só COs pode gerenciar usuários. O HSM contém um CO padrão chamado admin. Você define a senha para admin quando [ativou o cluster](#).

Este tópico fornece step-by-step instruções e detalhes sobre o gerenciamento de usuários do HSM com o AWS CloudHSM Management Utility (CMU).

Tópicos

- [Pré-requisitos para gerenciamento de usuários no Management Utility AWS CloudHSM](#)
- [Tipos de usuário do HSM para o AWS CloudHSM Management Utility](#)
- [Tabela de permissões de usuário do HSM para o AWS CloudHSM Management Utility](#)
- [Crie usuários do HSM usando o AWS CloudHSM Management Utility](#)

- [Listar todos os usuários do HSM no cluster usando o AWS CloudHSM Management Utility](#)
- [Altere as senhas de usuário do HSM usando o AWS CloudHSM Management Utility](#)
- [Excluir usuários do HSM usando o AWS CloudHSM Management Utility](#)
- [Gerencie 2FA para usuários usando o AWS CloudHSM Management Utility](#)
- [Como usar o CloudHSM Management Utility \(CMU\) para gerenciar a autenticação de quórum \(controle de acesso M ou N\)](#)

Pré-requisitos para gerenciamento de usuários no Management Utility AWS CloudHSM

Antes de usar o AWS CloudHSM Management Utility (CMU) para gerenciar usuários do módulo de segurança de hardware (HSM) em AWS CloudHSM, você deve preencher esses pré-requisitos. Os tópicos a seguir descrevem como começar a usar o CMU.

Seções

- [Obtenha o endereço IP de um HSM em AWS CloudHSM](#)
- [Usando o CMU com o Client SDK 3.2.1 e versões anteriores](#)
- [Fazer o download de CloudHSM Management Utility](#)

Obtenha o endereço IP de um HSM em AWS CloudHSM

Para usar o CMU, você deve usar a ferramenta de configuração para atualizar a configuração local. O CMU cria sua própria conexão com o cluster e essa conexão não reconhece o cluster. Para rastrear as informações do cluster, o CMU mantém um arquivo de configuração local. Isso significa que toda vez que você usa o CMU, você deve primeiro atualizar o arquivo de configuração executando a ferramenta de linha de comando [configure](#) com o `--cmu` parâmetro. Se você estiver usando o Client SDK 3.2.1 ou anterior, deverá usar um parâmetro diferente de `--cmu`. Para obter mais informações, consulte [the section called “Usando o CMU com o Client SDK 3.2.1 e versões anteriores”](#).

O parâmetro `--cmu` exige que você adicione o endereço IP de um HSM no seu cluster. Se você tiver vários HSMs, poderá usar qualquer endereço IP. Isso garante que o CMU possa propagar todas as alterações que você fizer em todo o cluster. Lembre-se de que o CMU usa seu arquivo local para rastrear as informações do cluster. Se o cluster mudou desde a última vez em que você usou o

CMU de um host específico, você deve adicionar essas alterações ao arquivo de configuração local armazenado nesse host. Nunca adicione ou remova um HSM enquanto estiver usando o CMU.

Para obter um endereço IP para um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. Para abrir a página de detalhes do cluster, na tabela do cluster, escolha o ID do cluster.
4. Para obter o endereço IP, vá até a HSMs guia. Para IPv4 clusters, escolha um endereço listado em IPv4 Endereço ENI. Para clusters de pilha dupla, use o endereço ENI IPv4 ou ENI. IPv6

Para obter um endereço IP para um HSM (AWS CLI)

- Obtenha o endereço IP de um HSM usando o comando [describe-clusters](#) do AWS CLI. Na saída do comando, o endereço IP do HSMs são os valores de `EniIp` e `EniIPv6` (se for um cluster de pilha dupla).

```
$ aws cloudhsmv2 describe-clusters
{
  "Clusters": [
    { ... }
    "Hsms": [
      {
...
        "EniIp": "10.0.0.9",
...
      },
      {
...
        "EniIp": "10.0.1.6",
        "EniIPv6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
      }
    ]
  }
}
```

Usando o CMU com o Client SDK 3.2.1 e versões anteriores

Com o Client SDK 3.3.0, AWS CloudHSM foi adicionado suporte ao `--cmu` parâmetro, o que simplifica o processo de atualização do arquivo de configuração para CMU. Se você estiver usando uma versão do CMU do Client SDK 3.2.1 ou anterior, deverá continuar usando `-m` os parâmetros -

a e para atualizar o arquivo de configuração. Para obter mais informações sobre estes parâmetros, consulte [Configurar ferramenta](#).

Fazer o download de CloudHSM Management Utility

A versão mais recente do CMU está disponível para tarefas de gerenciamento de usuários do HSM, independentemente de você estar usando o Client SDK 5 e o Client SDK 3.

Para fazer o download e instalar o CMU

- Faça download e instale o CMU.

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-mgmt-util-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

CentOS 7.8+

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

CentOS 8.3+

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-mgmt-util-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el8.x86_64.rpm
```

RHEL 7 (7.8+)

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

RHEL 8 (8.3+)

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-mgmt-util-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-mgmt-util_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-mgmt-util_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-mgmt-util_latest_u18.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-mgmt-util_latest_u18.04_amd64.deb
```

Windows Server 2012

1. Faça o download do [CloudHSM Management Utility](#).
2. Execute o instalador CMU (AWSCloudHSMManagementUtil-latest.msi) com privilégios administrativos do Windows.

Windows Server 2012 R2

1. Faça o download do [CloudHSM Management Utility](#).
2. Execute o instalador CMU (AWSCloudHSMManagementUtil-latest.msi) com privilégios administrativos do Windows.

Windows Server 2016

1. Faça o download do [CloudHSM Management Utility](#).
2. Execute o instalador CMU (AWSCloudHSMManagementUtil-latest.msi) com privilégios administrativos do Windows.

Tipos de usuário do HSM para o AWS CloudHSM Management Utility

A maioria das operações que você executa no módulo de segurança de hardware (HSM) exige as credenciais de um usuário do AWS CloudHSM HSM. O HSM autentica cada usuário do HSM e cada usuário do HSM tem um tipo que determina quais operações você pode realizar no HSM como esse usuário.

Note

Os usuários do HSM são diferentes dos usuários do IAM. Os usuários do IAM que têm as credenciais corretas podem criar interagindo com recursos HSMs por meio da API da AWS. Depois que o HSM for criado, você deverá usar as credenciais de usuário do HSM para autenticar as operações no HSM.

Tipos de usuário

- [Responsável pela pré-criptografia \(PRECO\)](#)
- [Responsável pela criptografia \(CO\)](#)
- [Crypto user \(CU\) \(usuário de criptografia\)](#)
- [Usuário de dispositivo \(AU\)](#)

Responsável pela pré-criptografia (PRECO)

Tanto no utilitário de gerenciamento de nuvem (CMU) quanto no utilitário de gerenciamento de chaves (KMU), o PRECO é um usuário temporário que existe somente no primeiro HSM em um cluster AWS CloudHSM. O primeiro HSM em um novo cluster contém um usuário PRECO indicando que esse cluster nunca foi ativado. Para [ativar um cluster](#), você executa a `cloudhsm-cli` e executa o comando `cluster activate`. Faça login no HSM e altere a senha do PRECO. Ao alterar a senha, o usuário PRECO se torna um responsável pela criptografia (CO).

Responsável pela criptografia (CO)

Tanto no utilitário de gerenciamento de nuvem (CMU) quanto no utilitário de gerenciamento de chaves (KMU), um responsável pela criptografia (CO) pode realizar operações de gerenciamento de usuários. Por exemplo, eles podem criar e excluir usuários e alterar suas senhas. Para obter mais informações sobre usuários CO, consulte [Tabela de permissões de usuário do HSM para o AWS CloudHSM Management Utility](#). Quando você ativa um novo cluster, o usuário muda de [Precrypto Officer \(PRECO\)](#) para [Crypto Officer \(CO\)](#).

Crypto user (CU) (usuário de criptografia)

Um usuário de criptografia (CU) pode realizar as seguintes operações de criptografia e gerenciamento de chaves.

- Gerenciamento de chaves: criar, excluir, compartilhar, importar e exportar chaves criptográficas.
- Operações criptográficas: use chaves criptográficas para criptografia, descryptografia, assinatura, verificação e muito mais.

Para obter mais informações, consulte o [Tabela de permissões de usuário do HSM para o AWS CloudHSM Management Utility](#).

Usuário de dispositivo (AU)

O usuário do equipamento (AU) pode realizar operações de clonagem e sincronização no seu cluster. HSMs AWS CloudHSM usa a AU para sincronizar o HSMs em um AWS CloudHSM cluster. O AU existe em todos os HSMs fornecidos por AWS CloudHSM, e tem permissões limitadas. Para obter mais informações, consulte o [Tabela de permissões de usuário do HSM para o AWS CloudHSM Management Utility](#).

AWS não pode realizar nenhuma operação no seu HSMs . AWS não pode visualizar ou modificar seus usuários ou chaves e não pode realizar nenhuma operação criptográfica usando essas chaves.

Tabela de permissões de usuário do HSM para o AWS CloudHSM Management Utility

A tabela a seguir lista as operações do HSM classificadas pelo tipo de usuário ou sessão do HSM que pode realizar a operação no AWS CloudHSM.

	Responsável pela criptografia (CO)	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Obter informações ¹ básicas do cluster	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)
Alterar a própria senha	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	Não aplicável
Alterar a senha de qualquer usuário	 Sim	 No (Não)	 Não	 Não
Adicionar , remover usuários	 Sim	 No (Não)	 Não	 Não
Obter status ³ de sincronização	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)

	Responsável pela criptografia (CO)	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Extrair, inserir objetos mascarados ⁴	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Funções de gerenciamento de chaves ⁵	 No (Não)	 Sim	 No (Não)	 Não
Criptografar, descriptografar	 No (Não)	 Sim	 No (Não)	 Não
Assinar, verificar	 No (Não)	 Sim	 No (Não)	 Não
Gere resumos e HMACs	 No (Não)	 Sim	 No (Não)	 Não

- [1] As informações básicas do cluster incluem o número de pessoas HSMs no cluster e o endereço IP, modelo, número de série, ID do dispositivo, ID do firmware, etc.
- [2] O usuário pode obter um conjunto de arquivos de resumo (hashes) que correspondem às chaves no HSM. Um aplicativo pode comparar esses conjuntos de resumos para entender o status de sincronização HSMs em um cluster.
- [3] Objetos mascarados são chaves criptografadas antes de sair do HSM. Elas não podem ser descriptografadas fora do HSM. Somente são descriptografadas depois de serem inseridas em um

HSM que esteja no mesmo cluster que o HSM do qual foram extraídas. Um aplicativo pode extrair e inserir objetos mascarados para sincronizá-los HSMs em um cluster.

- [4] As funções de gerenciamento de chaves incluem criar, excluir, encapsular, desencapsular e modificar os atributos das chaves.

Crie usuários do HSM usando o AWS CloudHSM Management Utility

Use `createUser` no Utilitário AWS CloudHSM de Gerenciamento (CMU) para criar novos usuários no módulo de segurança de hardware (HSM). Você deve fazer login como CO para criar um usuário.

Para criar um novo usuário CO

1. Use a ferramenta de configuração para atualizar a configuração do CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Iniciar o CMU.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\  
\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Faça login no console do HSM como usuário CO.

```
aws-cloudhsm > loginHSM CO admin co12345
```

Certifique-se de que o número de conexões das listas CMU corresponda ao número de conexões HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use `createUser` para criar um usuário de CO chamado **example_officer** com uma senha de **password1**.

```
aws-cloudhsm > createUser CO example_officer password1
```

O CMU avisa sobre a operação de criação de usuário.

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?
```

5. Digite **y**.

Para criar um novo usuário CU

1. Use a ferramenta de configuração para atualizar a configuração do CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Iniciar o CMU.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Faça login no console do HSM como usuário CO.

```
aws-cloudhsm > loginHSM CO admin co12345
```

Certifique-se de que o número de conexões das listas CMU corresponda ao número de conexões HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use `createUser` para criar um usuário de CU chamado **example_user** com uma senha de **password1**.

```
aws-cloudhsm > createUser CU example_user password1
```

O CMU avisa sobre a operação de criação de usuário.

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****
Do you want to continue(y/n)?
```

5. Digite **y**.

Para obter mais informações sobre `createUser`, consulte [CreateUser](#).

Listar todos os usuários do HSM no cluster usando o AWS CloudHSM Management Utility

Use o `listUsers` comando no Utilitário de AWS CloudHSM Gerenciamento (CMU) para listar todos os usuários no AWS CloudHSM cluster. Você não precisa fazer login para executar `listUsers` e todos os tipos de usuário podem listar usuários.

Para listar todos os usuários no cluster

1. Use a ferramenta de configuração para atualizar a configuração do CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Iniciar o CMU.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Use listUsers para listar todos os usuários no cluster.

```
aws-cloudhsm > listUsers
```

O CMU lista todos os usuários no cluster.

Users on server 0(10.0.2.9):

Number of users found:4

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	AU	app_user	NO
2	CO	example_officer	NO
3	CU	example_user	NO

Users on server 1(10.0.3.11):


```
Number of users found:4
```

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	AU	app_user	N0
	0		N0
2	CO	example_officer	N0
	0		N0
3	CU	example_user	N0
	0		N0

```
Users on server 2(10.0.1.12):
```

```
Number of users found:4
```

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	AU	app_user	N0
	0		N0
2	CO	example_officer	N0
	0		N0
3	CU	example_user	N0
	0		N0

Para obter mais informações sobre listUsers, consulte [listUsers](#).

Altere as senhas de usuário do HSM usando o AWS CloudHSM Management Utility

Use changePswd no Utilitário AWS CloudHSM de Gerenciamento (CMU) para alterar a senha de um usuário do módulo de segurança de hardware (HSM).

Os tipos de usuários e as senhas diferenciam maiúsculas e minúsculas, mas os nomes de usuários não.

CO, usuários de criptografia (CUs) e usuários do dispositivo (AUs) podem alterar suas próprias senhas. Para alterar a senha de outro usuário, você deve fazer login como um CO. Não é possível alterar a senha de um usuário que está conectado no momento.

Para alterar sua senha

1. Use a ferramenta de configuração para atualizar a configuração do CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Iniciar o CMU.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Fazer login em HSMs.

```
aws-cloudhsm > loginHSM C0 admin co12345
```

Certifique-se de que o número de conexões das listas CMU corresponda ao número de conexões HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use changePswd para alterar sua senha.

```
aws-cloudhsm > changePswd C0 example_officer <new password>
```

O CMU solicita a operação de alteração de senha.

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****
```

```
Do you want to continue(y/n)?
```

5. Digite **y**.

O CMU solicita a operação de alteração de senha.

```
Changing password for example_officer(C0) on 3 nodes
```

Para alterar a senha de outro usuário

1. Use a ferramenta de configuração para atualizar a configuração do CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Iniciar o CMU.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Faça login no console do HSM como usuário CO.

```
aws-cloudhsm > loginHSM C0 admin co12345
```

Certifique-se de que o número de conexões das listas CMU corresponda ao número de conexões HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use changePswd para alterar a senha de outro usuário.

```
aws-cloudhsm > changePswd CU example_user <new password>
```

O CMU solicita a operação de alteração de senha.

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?
```

5. Digite **y**.

O CMU solicita a operação de alteração de senha.

```
Changing password for example_user(CU) on 3 nodes
```

Para obter mais informações sobre o `changePswd`, consulte [changePswd](#).

Excluir usuários do HSM usando o AWS CloudHSM Management Utility

Use `deleteUser` no Utilitário AWS CloudHSM de Gerenciamento (CMU) para excluir um usuário do módulo de segurança de hardware (HSM). Você deve fazer login como CO para excluir outro usuário.



Você não pode excluir usuários de criptografia (CU) que possuem chaves.

Para excluir um usuário

1. Use a ferramenta de configuração para atualizar a configuração do CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Iniciar o CMU.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Faça login no console do HSM como usuário CO.

```
aws-cloudhsm > loginHSM C0 admin co12345
```

Certifique-se de que o número de conexões das listas CMU corresponda ao número de conexões HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use deleteUser para excluir um usuário.

```
aws-cloudhsm > deleteUser C0 example_officer
```

A CMU exclui o usuário.

```
Deleting user example_officer(C0) on 3 nodes
deleteUser success on server 0(10.0.2.9)
deleteUser success on server 1(10.0.3.11)
deleteUser success on server 2(10.0.1.12)
```

Para obter mais informações sobre deleteUser, consulte [deleteUser](#).

Gerencie 2FA para usuários usando o AWS CloudHSM Management Utility

Para aumentar a segurança, você pode configurar a autenticação de dois fatores (2FA) para ajudar a proteger o cluster do AWS CloudHSM. Você só pode ativar o 2FA para responsáveis pela criptografia (CO).

Ao fazer login em um cluster com uma conta de módulo de serviço de hardware (HSM) habilitada para 2FA, você fornece sua senha ao cloudhsm_mgmt_util (CMU), o primeiro fator, o que você sabe, e a CMU fornece um token e solicita que você assine o token. Para fornecer o segundo fator, o que você tem, você assina o token com uma chave privada de um par de chaves que já criou e associou ao usuário do HSM. Para acessar o cluster, você fornece o token assinado à CMU.

Note

Você não pode ativar a 2FA para usuários de criptografia (CU) ou aplicativos. A autenticação de dois fatores (2FA) é somente para usuários de CO.

Tópicos

- [Autenticação de quorum e 2FA em AWS CloudHSM clusters usando o Management Utility AWS CloudHSM](#)
- [Requisitos de pares de chaves 2FA para AWS CloudHSM usar o AWS CloudHSM Management Utility](#)
- [Crie usuários com 2FA habilitado para usuários do AWS CloudHSM Management Utility](#)
- [Gerencie 2FA para usuários de HSM usando o AWS CloudHSM Management Utility](#)
- [Desative a 2FA para usuários do HSM usando o AWS CloudHSM Management Utility](#)
- [Referência de configuração para 2FA com o AWS CloudHSM Management Utility](#)

Autenticação de quorum e 2FA em AWS CloudHSM clusters usando o Management Utility AWS CloudHSM

O cluster usa a mesma chave para autenticação de quórum e para autenticação de dois fatores (2FA). Isso significa que um usuário com 2FA ativado está efetivamente registrado para M-of-N-access -control (MoFN). Para usar com sucesso a autenticação 2FA e de quórum para o mesmo usuário do HSM, considere os seguintes pontos:

- Se estiver usando a autenticação de quórum para um usuário hoje, você deve usar o mesmo par de chaves que criou para o usuário do quórum para habilitar a 2FA para ele.
- Se você adicionar o requisito de 2FA para um usuário não 2FA que não seja um usuário de autenticação de quorum, registre esse usuário como um usuário MoFN com autenticação 2FA.
- Se você remover o requisito de 2FA ou alterar a senha de um usuário de 2FA que também seja usuário de autenticação de quórum, também removerá o registro do usuário do quórum como usuário do MoFN.
- Se você remover o requisito de 2FA ou alterar a senha de um usuário de 2FA que também é usuário de autenticação de quórum, mas ainda quiser que esse usuário participe da autenticação de quórum, deverá registrá-lo novamente como usuário do MoFN.

Para obter mais informações sobre a autenticação de quórum, consulte [Como usar o CMU para gerenciar a autenticação de quórum](#).

Requisitos de pares de chaves 2FA para AWS CloudHSM usar o AWS CloudHSM Management Utility

Para habilitar a autenticação de dois fatores (2FA) para um usuário do módulo de segurança de AWS CloudHSM hardware (HSM), use uma chave que atenda aos seguintes requisitos.

Você pode criar um novo par de chaves ou usar uma chave existente que atenda aos seguintes requisitos.

- Tipo de chave: assimétrico
- Uso da chave: assinar e verificar
- Especificação da chave: RSA_2048
- O algoritmo de assinatura inclui:
 - sha256WithRSAEncryption

Note

Se você estiver usando a autenticação de quórum ou planeja usar a autenticação de quórum, consulte. [the section called “Autenticação de quórum”](#)

Crie usuários com 2FA habilitado para usuários do AWS CloudHSM Management Utility

Use o AWS CloudHSM Management Utility CMU (CMU) e o key pair para criar um novo usuário do crypto office (CO) com a autenticação de dois fatores (2FA) ativada.

Para criar usuários de CO com 2FA habilitado

1. Em um terminal, execute as seguintes etapas:

a. Acesse seu HSM e faça login no utilitário de gerenciamento do CloudHSM:

```
/opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

b. Faça login como CO e use o comando a seguir para criar um novo usuário do MFA com 2FA:

```
aws-cloudhsm > createUser CO MFA <CO USER NAME> -2fa /home/ec2-user/authdata
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)? y

Creating User exampleuser3(CO) on 1 nodesAuthentication data written to: "/
home/ec2-user/authdata"Generate Base64-encoded signatures for SHA256 digests in
the authentication datafile.
To generate the signatures, use the RSA private key, which is the second factor
ofauthentication for this user. Paste the signatures and the corresponding
public keyinto the authentication data file and provide
the file path below.Leave this field blank to use the path initially
provided.Enter filename:
```

c. Deixe o terminal acima nesse estado. Não pressione enter nem insira nenhum nome de arquivo.

2. Em outro terminal, execute as seguintes etapas:

a. Acesse seu HSM e faça login no utilitário de gerenciamento do CloudHSM:


```
/opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

- b. Gere um par de chaves público-privado usando os seguintes comandos:

```
openssl genpkey -algorithm RSA -out private_key.pem -pkeyopt  
rsa_keygen_bits:2048
```

```
openssl rsa -pubout -in private_key.pem -out public_key.pem
```

- c. Execute o comando a seguir para instalar um recurso de consulta json para extrair o resumo do arquivo authdata:

```
sudo yum install jq
```

- d. Para extrair o valor do resumo, primeiro encontre os seguintes dados no arquivo authdata:

```
{  
  "Version": "1.0",  
  "PublicKey": "",  
  "Data": [  
    {  
      "HsmId": <"HSM ID">,  
      "Digest": <"DIGEST">,  
      "Signature": ""  
    }  
  ]  
}
```

Note

O resumo obtido é codificado em base64. No entanto, para assinar o resumo, primeiro você precisa que o arquivo seja decodificado e depois assinado. O comando a seguir decodificará o resumo e armazenará o conteúdo decodificado em “digest1.bin”

```
cat authdata | jq '.Data[0].Digest' | cut -c2- | rev | cut -c2- | rev |  
base64 -d > digest1.bin
```

- e. Converta o conteúdo da chave pública, adicionando "\n" e removendo espaços conforme mostrado aqui:

```
-----BEGIN PUBLIC KEY-----\n<PUBLIC KEY>\n-----END PUBLIC KEY-----
```

Important

O comando acima mostra como "\n" é adicionado imediatamente depois de BEGIN PUBLIC KEY-----, os espaços entre "\n" e o primeiro caractere da chave pública são removidos, "\n" é adicionado antes -----END PUBLIC KEY e os espaços são removidos entre "\n" e o final da chave pública.

Esse é o formato PEM para chave pública aceito no arquivo authdata.

- f. Cole o conteúdo do formato pem da chave pública na seção de chave pública no arquivo authdata.

```
vi authdata
```

```
{
  "Version": "1.0",
  "PublicKey": "-----BEGIN PUBLIC KEY-----\n<PUBLIC KEY>\n-----END PUBLIC KEY-----",
  "Data": [
    {
      "HsmId": "<HSM ID>",
      "Digest": "<DIGEST>",
      "Signature": ""
    }
  ]
}
```

- g. Assine um arquivo de token usando o seguinte comando:

```
openssl pkeyutl -sign -in digest1.bin -inkey private_key.pem -pkeyopt
digest:sha256 | base64
```

Output Expected:

```
<THE SIGNATURE>
```

Note

Conforme mostrado no comando acima, use `openssl pkeyutl` em vez de `openssl dgst` para assinar.

- h. Adicione o resumo assinado no arquivo Authdata no campo “Assinatura”.

```
vi authdata
```

```
{
  "Version": "1.0",
  "PublicKey": "-----BEGIN PUBLIC KEY----- ... -----END PUBLIC KEY-----",
  "Data": [
    {
      "HsmId": <"HSM ID">,
      "Digest": <"DIGEST">,
      "Signature": <"Kkd1 ... rkrvJ6Q==">
    },
    {
      "HsmId": <"HSM ID">,
      "Digest": <"DIGEST">,
      "Signature": <"K1hxy ... Q261Q==">
    }
  ]
}
```

3. Volte para o primeiro terminal e pressione **Enter**:

Generate Base64-encoded signatures for SHA256 digests in the authentication datafile. To generate the signatures, use the RSA private key, which is the second factor of authentication for this user. Paste the signatures and the corresponding public key into the authentication data file and provide the file path below. Leave this field blank to use the path initially provided.

Enter filename: >>>> Press Enter here

```
createUser success on server 0(10.0.1.11)
```

Gerencie 2FA para usuários de HSM usando o AWS CloudHSM Management Utility

Use `changePswd` no Utilitário AWS CloudHSM de Gerenciamento (CMU) para modificar a autenticação de dois fatores (2FA) para um usuário. Cada vez que você habilita a 2FA, deve fornecer uma chave pública para logins de 2FA.

O comando `changePswd` executa um ou todos os seguintes cenários:

- Alterar a senha para um usuário de 2FA
- Alterar a senha para um usuário que não seja 2FA
- Adicionar 2FA a um usuário que não seja 2FA
- Remover 2FA de um usuário de 2FA
- Rotacionar a chave para um usuário de 2FA

Você também pode combinar tarefas. Por exemplo, você pode remover a 2FA de um usuário e alterar a senha ao mesmo tempo, ou pode alternar a chave 2FA e alterar a senha do usuário.

Para alterar senhas ou alternar chaves para usuários de CO com 2FA habilitado

1. Use o CMU para fazer login no HSM como um CO com 2FA ativado.
2. Use `changePswd` para alterar a senha ou alternar a chave de usuários de CO com 2FA habilitado. Use o `-2fa` parâmetro e inclua um local no sistema de arquivos para que o sistema grave o `authdata` arquivo. Esse arquivo inclui um resumo para cada HSM no cluster.

```
aws-cloudhsm > changePswd CO example-user <new-password> -2fa /path/to/authdata
```

O CMU solicita que você use a chave privada para assinar os resumos no `authdata` arquivo e retornar as assinaturas com a chave pública.

3. Use a chave privada para assinar os resumos no `authdata` arquivo, adicionar as assinaturas e a chave pública ao `authdata` arquivo formatado em JSON e, em seguida, fornecer à CMU a localização do `authdata` arquivo. Para obter mais informações, consulte [the section called “Referência da configuração”](#).

Note

O cluster usa a mesma chave para autenticação de quórum e de 2FA. Se você estiver usando a autenticação de quórum ou planeja usar a autenticação de quórum, consulte [the section called “Autenticação de quórum”](#)

Desative a 2FA para usuários do HSM usando o AWS CloudHSM Management Utility

Use o Utilitário AWS CloudHSM de Gerenciamento (CMU) para desativar a autenticação de dois fatores (2FA) para usuários do módulo de segurança de hardware (HSM) em AWS CloudHSM

Para desativar 2FA para usuários de CO com 2FA ativado

1. Use o CMU para fazer login no HSM como um CO com 2FA ativado.
2. Use `changePswd` para remover 2FA de usuários de CO com 2FA habilitado.

```
aws-cloudhsm > changePswd CO example-user <new password>
```

A CMU solicita a operação de alteração de senha.

Note

Se você remover o requisito de 2FA ou alterar a senha de um usuário de 2FA que também seja usuário de autenticação de quórum, também removerá o registro do usuário do quórum como usuário do MoFN. Para obter mais informações sobre usuários do quórum e 2FA, consulte [the section called “Autenticação de quórum”](#).

3. Digite `y`.

A CMU confirma a operação de alteração de senha.

Referência de configuração para 2FA com o AWS CloudHSM Management Utility

Veja a seguir um exemplo das propriedades da autenticação de dois fatores (2FA) no `authdata` arquivo para a solicitação gerada pelo AWS CloudHSM Management Utility (CMU) e para suas respostas.

```
{
  "Version": "1.0",
  "PublicKey": "-----BEGIN PUBLIC KEY----- ... -----END PUBLIC KEY-----",
  "Data": [
    {
      "HsmId": "hsm-lgavqitns2a",
      "Digest": "k501p3f6foQRVQH7S8Rrjcau6h3TYqsSdr16A54+qG8=",
      "Signature": "Kkd1 ... rkrvJ6Q=="
    },
    {
      "HsmId": "hsm-lgavqitns2a",
      "Digest": "IyBcx4I5Vyx1jztwvXinCBQd9lDx8oQe7iRrWjBAi1w=",
      "Signature": "K1hxy ... Q261Q=="
    }
  ]
}
```

Dados

Nó de nível superior. Contém um nó subordinado para cada HSM no cluster. Aparece nas solicitações e respostas de todos os comandos 2FA.

Resumo

Isso é o que você deve assinar para fornecer o segundo fator de autenticação. CMU gerada em solicitações para todos os comandos 2FA.

HsmId

O ID do seu HSM. Aparece nas solicitações e respostas de todos os comandos 2FA.

PublicKey

A parte da chave pública do par de chaves que você gerou foi inserida como string formatada em PEM. Você insere isso nas respostas para createUser e changePswd.

Assinatura

O resumo assinado codificado em Base 64. Você insere isso nas respostas para todos os comandos 2FA.

Version (Versão)

A versão do arquivo dos dados de autenticação formatado em JSON. Aparece nas solicitações e respostas de todos os comandos 2FA.

Como usar o CloudHSM Management Utility (CMU) para gerenciar a autenticação de quórum (controle de acesso M ou N)

O HSMs em seu AWS CloudHSM cluster oferece suporte à autenticação de quorum, que também é conhecida como controle de acesso M of N. Com a autenticação de quorum, nenhum usuário único no HSM pode fazer operações controladas pelo quorum no HSM. Em vez disso, um número mínimo de usuários do HSM (pelo menos 2) deve cooperar para realizar essas operações. Com a autenticação de quorum, você pode adicionar uma camada adicional de proteção, exigindo aprovação de mais de um usuário do HSM.

A autenticação de quorum pode controlar as seguintes operações:

- Gerenciamento de usuários do HSM por [agentes de criptografia \(COs\)](#) — Criação e exclusão de usuários do HSM e alteração da senha de um usuário do HSM diferente. Para obter mais informações, consulte [Gerenciamento de usuários com autenticação de quorum habilitada para AWS CloudHSM o Management Utility](#).

Observe as seguintes informações adicionais sobre como usar a autenticação de quorum no AWS CloudHSM.

- Um usuário do HSM pode assinar seu próprio token de quorum, ou seja, o usuário solicitante pode fornecer uma das aprovações necessárias para a autenticação de quorum.
- Você escolhe o número mínimo de aprovadores de quorum para operações controladas por quorum. O menor número que é possível escolher é dois (2) e o maior número que é possível escolher é oito (8).
- O HSM pode armazenar até 1024 tokens de quorum. Se o HSM já tiver 1024 tokens quando você tentar criar um novo, ele removerá um dos tokens expirados. Por padrão, tokens expiram dez minutos após sua criação.
- O cluster usa a mesma chave para autenticação de quórum e para autenticação de dois fatores (2FA). Para obter mais informações sobre como usar a autenticação de quórum e a 2FA, consulte [Autenticação de quórum e 2FA](#).

Os seguintes tópicos fornecem mais informações sobre a autenticação de quorum no AWS CloudHSM.

Tópicos

- [Processo de autenticação de quorum para AWS CloudHSM o Management Utility](#)
- [Configure a autenticação de quórum para agentes de criptografia AWS CloudHSM](#)
- [Gerenciamento de usuários com autenticação de quorum habilitada para AWS CloudHSM o Management Utility](#)
- [Altere o valor mínimo do quórum com AWS CloudHSM o Management Utility](#)

Processo de autenticação de quorum para AWS CloudHSM o Management Utility

As etapas a seguir resumem os processos de autenticação de quorum. Para conhecer as etapas e ferramentas específicas, consulte [Gerenciamento de usuários com autenticação de quorum habilitada para AWS CloudHSM o Management Utility](#).

1. Cada usuário do HSM cria uma chave assimétrica para assinatura. Isso é feito fora do HSM, tomando cuidado para proteger a chave apropriadamente.
2. Cada usuário do HSM faz login no HSM e registra a parte pública de sua chave de assinatura (a chave pública) no HSM.
3. Quando um usuário do HSM quer fazer uma operação controlada por quorum, ele faz login no HSM e obtém um token de quorum.
4. O usuário do HSM fornece esse token de quorum a um ou mais usuários do HSM e solicita sua aprovação.
5. Os outros usuários do HSM aprovam usando suas chaves para assinar criptograficamente o token de quorum. Isso ocorre fora do HSM.
6. Quando o usuário do HSM tiver o número de aprovações necessárias, ele fará login no HSM e fornecerá o token de quorum e as aprovações (assinaturas) ao HSM.
7. O HSM usa as chaves públicas registradas de cada assinante para verificar as assinaturas. Se as assinaturas forem válidas, o HSM aprovará o token.
8. O usuário do HSM agora pode fazer uma operação controlada por quorum.

Configure a autenticação de quórum para agentes de criptografia AWS CloudHSM

Os tópicos a seguir descrevem as etapas que você deve concluir para configurar seu módulo de segurança de hardware (HSM) para que [os agentes de AWS CloudHSM criptografia \(COs\)](#) possam usar a autenticação de quorum. Você precisa executar essas etapas somente uma vez ao configurar a autenticação de quorum pela primeira vez para COs. Depois de concluir essas etapas,

consulte [Gerenciamento de usuários com autenticação de quorum habilitada para AWS CloudHSM o Management Utility](#).

Tópicos

- [Pré-requisitos](#)
- [Etapa 1. Criar e registrar uma chave para assinatura](#)
- [Etapa 2. Definir o valor mínimo do quorum no HSM](#)

Pré-requisitos

Para entender esse exemplo, você deve estar familiarizado com a [ferramenta da linha de comando do CMUcloudhsm_mgmt_util](#). Neste exemplo, o AWS CloudHSM cluster tem dois HSMs, cada um com o mesmo COs, conforme mostrado na saída a seguir do listUsers comando. Para obter mais informações sobre como criar usuários, consulte [Usuários do HSM](#).

```
aws-cloudhsm > listUsers
```

```
Users on server 0(10.0.2.14):
```

```
Number of users found:7
```

User Id LoginFailureCnt	User Type 2FA	User Name	MofnPubKey
1	PRECO	admin	N0
0	N0		
2	AU	app_user	N0
0	N0		
3	CO	officer1	N0
0	N0		
4	CO	officer2	N0
0	N0		
5	CO	officer3	N0
0	N0		
6	CO	officer4	N0
0	N0		
7	CO	officer5	N0
0	N0		

```
Users on server 1(10.0.1.4):
```

```
Number of users found:7
```

User Id LoginFailureCnt	User Type 2FA	User Name	MofnPubKey

1	PRECO	admin	NO
0	NO		
2	AU	app_user	NO
0	NO		
3	CO	officer1	NO
0	NO		
4	CO	officer2	NO
0	NO		
5	CO	officer3	NO
0	NO		
6	CO	officer4	NO
0	NO		
7	CO	officer5	NO
0	NO		

Etapa 1. Criar e registrar uma chave para assinatura

Para usar a autenticação de quórum, cada CO deve executar todas as etapas a seguir:

Tópicos

- [Criar um par de chaves RSA](#)
- [Crie e assine um token de registro](#)
- [Registrar uma chave pública no HSM](#)

Criar um par de chaves RSA

Existem muitas maneiras diferentes de criar e proteger um par de chaves. O exemplo a seguir mostra como fazer isso com o [OpenSSL](#).

Exemplo Crie uma chave privada com o OpenSSL

O exemplo a seguir demonstra como usar o OpenSSL para criar uma chave RSA de 2048 bits protegida por um código de acesso. Para usar esse exemplo, *officer1.key* substitua pelo nome do arquivo em que você deseja armazenar a chave.

```
$ openssl genrsa -out <officer1.key> -aes256 2048
Generating RSA private key, 2048 bit long modulus
.....+++
.+++
e is 65537 (0x10001)
```

```
Enter pass phrase for officer1.key:  
Verifying - Enter pass phrase for officer1.key:
```

Em seguida, gere a chave pública usando a chave privada que você acabou de criar.

Example Crie uma chave pública com o OpenSSL

O exemplo a seguir demonstra como usar o OpenSSL para criar uma chave pública a partir da chave privada que você acabou de criar.

```
$ openssl rsa -in officer1.key -outform PEM -pubout -out officer1.pub  
Enter pass phrase for officer1.key:  
writing RSA key
```

Crie e assine um token de registro

Você cria um token e o assina com a chave privada que acabou de gerar na etapa anterior.

Example Crie um token

O token de registro é apenas um arquivo com dados randômicos que não excedem o tamanho máximo de 245 bytes. Você assina o token com a chave privada para demonstrar que tem acesso à chave privada. O comando a seguir usa echo para redirecionar uma string para um arquivo.

```
$ echo <token to be signed> > officer1.token
```

Assine o token e salve-o em um arquivo de assinatura. Você precisará do token assinado, do token não assinado e da chave pública para registrar o CO como usuário do MoFN no HSM.

Example Assine o token

Use o OpenSSL e a chave privada para assinar o token de registro e criar o arquivo de assinatura.

```
$ openssl dgst -sha256 \  
-sign officer1.key \  
-out officer1.token.sig officer1.token
```

Registrar uma chave pública no HSM

Depois de criar uma chave, o CO deve registrar a parte pública dela (a chave pública) no HSM.

Para registrar uma chave pública no HSM

1. Use o comando a seguir para iniciar a ferramenta de linha de comando `cloudhsm_mgmt_util`.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

2. Use o comando `loginHSM` para fazer login nos HSMs como CO. Para obter mais informações, consulte [???](#).
3. Use o comando [registerQuorumPubKey](#) para registrar a chave pública. Para obter mais informações, consulte o exemplo a seguir ou use o comando `help registerQuorumPubKey`.

Exemplo Registrar uma chave pública no HSM

O exemplo a seguir mostra como usar o comando `registerQuorumPubKey` na ferramenta da linha de comando `cloudhsm_mgmt_util` para registrar a chave pública de um CO no HSM. Para usar esse comando, o CO deve estar conectado no HSM. Substitua estes valores pelos seus próprios:

```
aws-cloudhsm > registerQuorumPubKey
CO <officer1> <officer1.token> <officer1.token.sig> <officer1.pub>
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)? y
registerQuorumPubKey success on server 0(10.0.2.14)
```

<officer1.token>

O caminho para um arquivo que contém um token de registro não assinado. Pode ter qualquer dado aleatório com tamanho máximo de arquivo de 245 bytes.

Obrigatório: Sim

<officer1.token.sig>

O caminho para um arquivo que contém o hash assinado pelo mecanismo SHA256 _PKCS do token de registro.

Obrigatório: Sim

<officer1.pub>

O caminho para o arquivo que contém a chave pública de um par de chaves assimétricas RSA-2048. Use a chave privada para assinar o token de registro.

Obrigatório: Sim

Depois de todos COs registrarem suas chaves públicas, a saída do listUsers comando mostra isso na MofnPubKey coluna, conforme mostrado no exemplo a seguir.

```
aws-cloudhsm > listUsers
```

```
Users on server 0(10.0.2.14):
```

```
Number of users found:7
```

User Id LoginFailureCnt	User Type 2FA	User Name	MofnPubKey
1	PRECO	admin	NO
0	NO		
2	AU	app_user	NO
0	NO		
3	CO	officer1	YES
0	NO		
4	CO	officer2	YES
0	NO		
5	CO	officer3	YES
0	NO		
6	CO	officer4	YES
0	NO		
7	CO	officer5	YES
0	NO		

```
Users on server 1(10.0.1.4):
```

```
Number of users found:7
```

User Id LoginFailureCnt	User Type 2FA	User Name	MofnPubKey
1	PRECO	admin	NO
0	NO		
2	AU	app_user	NO
0	NO		
3	CO	officer1	YES
0	NO		
4	CO	officer2	YES
0	NO		

5	CO	officer3	YES
0	NO		
6	CO	officer4	YES
0	NO		
7	CO	officer5	YES
0	NO		

Etapa 2. Definir o valor mínimo do quorum no HSM

Para usar a autenticação de quórum COs, um CO deve fazer login no HSM e definir o valor mínimo do quórum, também conhecido como valor m. Esse é o número mínimo de aprovações de CO necessárias para realizar operações de gerenciamento de usuários do HSM. Qualquer CO no HSM pode definir o valor mínimo do quórum, inclusive COs aquele que não registrou uma chave para assinatura. Você pode alterar o valor mínimo de quorum a qualquer momento. Para obter mais informações, consulte [Altere o valor mínimo](#).

Para definir o valor mínimo do quorum no HSM

1. Use o comando a seguir para iniciar a ferramenta de linha de comando cloudhsm_mgmt_util.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

2. Use o comando loginHSM para fazer login nos HSMs como CO. Para obter mais informações, consulte [???](#).
3. Use o comando setMValue para definir o valor mínimo do quorum. Para obter mais informações, consulte o exemplo a seguir ou use o comando help setMValue.

Example Definir o valor mínimo do quorum no HSM

Este exemplo usa um valor mínimo de quorum de dois. Você pode escolher qualquer valor de dois (2) a oito (8), até o número total de COs no HSM. Neste exemplo, o HSM tem seis COs, então o valor máximo possível é seis.

Para usar o comando de exemplo a seguir, substitua o número final (2) pelo valor mínimo de quórum preferido.

```
aws-cloudhsm > setMValue 3 <2>
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
```

```
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****
```

```
Do you want to continue(y/n)? y
Setting M Value(2) for 3 on 2 nodes
```

No exemplo anterior, o primeiro número (3) identifica o serviço HSM cujo valor mínimo de quorum você está configurando.

A tabela a seguir lista os identificadores de serviço do HSM junto com seus nomes, descrições e comandos incluídos no serviço.

Identificadores de dispositivo	Nome do serviço	Descrição do serviço	Comandos do HSM
3	USER_MGMT	Gerenciamento de usuários do HSM	• createUser • deleteUser • changePswd (aplica-se somente ao alterar a senha de outro usuário do HSM)
4	MISC_CO	Serviço de CO diverso	• setMValue

Para obter o valor mínimo de quorum para um serviço, use o comando `getMValue`, como no exemplo a seguir.

```
aws-cloudhsm > getMValue 3
MValue of service 3[USER_MGMT] on server 0 : [2]
MValue of service 3[USER_MGMT] on server 1 : [2]
```

A saída do comando `getMValue` anterior mostra que o valor mínimo de quorum para as operações de gerenciamento de usuários do HSM (serviço 3) é agora dois.

Depois de concluir essas etapas, consulte [Gerenciamento de usuários com autenticação de quorum habilitada para AWS CloudHSM o Management Utility](#).

Gerenciamento de usuários com autenticação de quorum habilitada para AWS CloudHSM o Management Utility

Um [oficial de AWS CloudHSM criptografia \(CO\)](#) no módulo de segurança de hardware (HSM) pode configurar a autenticação de quorum para as seguintes operações no HSM:

- Criar usuários do HSM
- Excluir usuários do HSM
- Alterar a senha de outro usuário do HSM

Depois que o HSM estiver configurado para autenticação de quórum, COs não poderá realizar as operações de gerenciamento de usuários do HSM por conta própria. O exemplo a seguir mostra a saída quando um CO tenta criar um novo usuário no HSM. O comando falha com um erro RET_MXN_AUTH_FAILED, o que indica que a autenticação de quorum falhou.

```
aws-cloudhsm > createUser CU user1 password
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)? y
Creating User user1(CU) on 2 nodes
createUser failed: RET_MXN_AUTH_FAILED
creating user on server 0(10.0.2.14) failed

Retry/Ignore/Abort?(R/I/A): A
```

Para realizar uma operação de gerenciamento de usuários do HSM, um CO deve completar as seguintes tarefas:

1. [Obtenha um token de quorum](#).
2. [Obtenha aprovações \(assinaturas\) de outros](#). COs
3. [Aprove o token no HSM](#).

4. [Realize a operação de gerenciamento de usuários do HSM.](#)

Se você ainda não configurou o HSM para autenticação de quorum COs, faça isso agora. Para obter mais informações, consulte [Configuração pela primeira vez](#).

Etapa 1. Obter um token de quorum

Primeiro, o CO deve usar a ferramenta da linha de comando `cloudhsm_mgmt_util` para solicitar um token de quorum.

Para obter um token de quorum

1. Use o comando a seguir para iniciar a ferramenta de linha de comando `cloudhsm_mgmt_util`.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

2. Use o comando `loginHSM` para fazer login nos HSMs como CO. Para obter mais informações, consulte [???](#).
3. Use o comando `getToken` para obter um token de quorum. Para obter mais informações, consulte o exemplo a seguir ou use o comando `help getToken`.

Example– Obter um token de quorum

Este exemplo obtém um token de quorum para o CO com o nome de usuário `officer1` e salva o token em um arquivo chamado `officer1.token`. Para usar o exemplo de comando, substitua esses valores por seus próprios:

- **`officer1`**— O nome do CO que está recebendo o token. Este deve ser o mesmo CO que está conectado ao HSM e executando esse comando.
- **`officer1.token`**— O nome do arquivo a ser usado para armazenar o token de quorum.

No comando a seguir, `3` identifica o serviço para o qual você pode usar o token que você está recebendo. Nesse caso, o token é para operações de gerenciamento de usuários do HSM (serviço 3). Para obter mais informações, consulte [Etapa 2. Definir o valor mínimo do quorum no HSM](#).

```
aws-cloudhsm > getToken 3 officer1 officer1.token
getToken success on server 0(10.0.2.14)
Token:
```

```

Id:1
Service:3
Node:1
Key Handle:0
User:officer1
getToken success on server 1(10.0.1.4)
Token:
Id:1
Service:3
Node:0
Key Handle:0
User:officer1

```

Etapa 2. Obtenha assinaturas ao aprovar COs

Um CO que tenha um token de quorum deve obter o token aprovado por outro. COs Para aprovar, o outro COs usa a chave de assinatura para assinar criptograficamente o token. Eles fazem isso fora do HSM.

Existem muitas maneiras diferentes de assinar o token. O exemplo a seguir mostra como fazer isso com o [OpenSSL](#). Para usar uma ferramenta de assinatura diferente, certifique-se de que essa ferramenta use a chave privada (chave de assinatura) do CO para assinar um resumo SHA-256 do token.

Example— Obtenha assinaturas ao aprovar COs

Neste exemplo, o CO que possui o token (officer1) precisa de pelo menos duas aprovações. Os comandos de exemplo a seguir mostram como dois COs podem usar o OpenSSL para assinar criptograficamente o token.

No primeiro comando, officer1 assina o seu próprio token. Para usar os seguintes comandos de exemplo, substitua esses valores por seus próprios:

- *officer1.key* e *officer2.key* — O nome do arquivo que contém a chave de assinatura do CO.
- *officer1.token.sig1* e *officer1.token.sig2* — O nome do arquivo a ser usado para armazenar a assinatura. Certifique-se de salvar cada assinatura em um arquivo diferente.
- *officer1.token* — O nome do arquivo que contém o token que o CO está assinando.

```
$ openssl dgst -sha256 -sign officer1.key -out officer1.token.sig1 officer1.token
```

```
Enter pass phrase for officer1.key:
```

No comando a seguir, officer2 assina o mesmo token.

```
$ openssl dgst -sha256 -sign officer2.key -out officer1.token.sig2 officer1.token
Enter pass phrase for officer2.key:
```

Etapa 3. Aprovar o token assinado no HSM

Depois que um CO obtém o número mínimo de aprovações (assinaturas) de outro COs, ele ou ela deve aprovar o token assinado no HSM.

Para aprovar o token assinado no HSM

1. Crie um arquivo de aprovação de token. Para obter mais informações, veja o exemplo a seguir.
2. Use o comando a seguir para iniciar a ferramenta de linha de comando cloudhsm_mgmt_util.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

3. Use o comando loginHSM para fazer login nos HSMs como CO. Para obter mais informações, consulte [???](#).
4. Use o comando approveToken para aprovar o token assinado, passando o arquivo de aprovação de token. Para obter mais informações, veja o exemplo a seguir.

Example Crie um arquivo de aprovação de token e aprove o token assinado no HSM

O arquivo de aprovação de token é um arquivo de texto em um formato específico exigido pelo HSM. O arquivo contém informações sobre o token, seus aprovadores e as assinaturas dos aprovadores. O seguinte mostra um exemplo de arquivo de aprovação de token.

```
# For "Multi Token File Path", type the path to the file that contains
# the token. You can type the same value for "Token File Path", but
# that's not required. The "Token File Path" line is required in any
# case, regardless of whether you type a value.
Multi Token File Path = officer1.token;
Token File Path = ;

# Total number of approvals
Number of Approvals = 2;
```

```
# Approver 1
# Type the approver's type, name, and the path to the file that
# contains the approver's signature.
Approver Type = 2; # 2 for CO, 1 for CU
Approver Name = officer1;
Approval File = officer1.token.sig1;

# Approver 2
# Type the approver's type, name, and the path to the file that
# contains the approver's signature.
Approver Type = 2; # 2 for CO, 1 for CU
Approver Name = officer2;
Approval File = officer1.token.sig2;
```

Depois de criar o arquivo de aprovação de token, o CO usa a ferramenta da linha de comando `cloudhsm_mgmt_util` para fazer login no HSM. O CO usa o comando `approveToken` para aprovar o token, como mostrado no exemplo a seguir. *approval.txt* Substitua pelo nome do arquivo de aprovação do token.

```
aws-cloudhsm > approveToken approval.txt
approveToken success on server 0(10.0.2.14)
approveToken success on server 1(10.0.1.4)
```

Quando esse comando é bem-sucedido, o HSM aprovou o token de quorum. Para verificar o status de um token, use o comando `listTokens`, conforme mostrado no exemplo a seguir. A saída do comando mostra que o token possui o número necessário de aprovações.

O tempo de validade de token indica quanto tempo o token é garantido para persistir no HSM. Mesmo após o período de validade do token (zero segundos), você ainda pode usá-lo.

```
aws-cloudhsm > listTokens
=====
      Server 0(10.0.2.14)
=====
----- Token - 0 -----
Token:
Id:1
Service:3
Node:1
Key Handle:0
User:officer1
```

```

Token Validity: 506 sec
Required num of approvers : 2
Current num of approvals : 2
Approver-0: officer1
Approver-1: officer2
Num of tokens = 1

=====
      Server 1(10.0.1.4)
=====
----- Token - 0 -----
Token:
Id:1
Service:3
Node:0
Key Handle:0
User:officer1
Token Validity: 506 sec
Required num of approvers : 2
Current num of approvals : 2
Approver-0: officer1
Approver-1: officer2
Num of tokens = 1

listTokens success

```

Etapa 4: Usar o token para operações de gerenciamento

Depois de um CO ter um token com o número necessário de aprovações, como mostrado na seção anterior, ele pode executar uma das seguintes operações de gerenciamento de usuários do HSM:

- Criar um usuário do HSM com o comando [createUser](#)
- Excluir um usuário do HSM com o comando `deleteUser`
- Alterar a senha de um usuário diferente do HSM com o comando `changePswd`

Para obter mais informações sobre como usar esses comandos, consulte [Usuários do HSM](#).

O CO pode usar o token para apenas uma operação. Quando essa operação for bem-sucedida, o token não será mais válido. Para fazer outra operação de gerenciamento de usuários do HSM, o CO deve obter um novo token de quorum, obter novas assinaturas dos aprovadores e aprovar o novo token no HSM.

Note

O token MofN só é válido enquanto sua sessão de login atual estiver aberta. Se você fizer log out do `cloudhsm_mgmt_util` ou se a conexão de rede for desconectada, o token não será mais válido. Da mesma forma, um token autorizado só pode ser usado em `cloudhsm_mgmt_util`. Ele não pode ser usado para autenticação em um aplicativo diferente.

No comando do exemplo a seguir, o CO cria um novo usuário no HSM.

```
aws-cloudhsm > createUser CU user1 <password>
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)? y
Creating User user1(CU) on 2 nodes
```

Depois que o comando anterior for bem-sucedido, um comando `listUsers` subsequente mostrará o novo usuário.

```
aws-cloudhsm > listUsers
Users on server 0(10.0.2.14):
Number of users found:8
```

User Id	User Type	User Name	MofnPubKey
1	PCO	admin	NO
0	2FA		
2	AU	app_user	NO
0	NO		
3	CO	officer1	YES
0	NO		
4	CO	officer2	YES
0	NO		
5	CO	officer3	YES
0	NO		

```

6          CO          officer4          YES
0          NO
7          CO          officer5          YES
0          NO
8          CU          user1             NO
0          NO

```

Users on server 1(10.0.1.4):

Number of users found:8

User Id LoginFailureCnt	User Type 2FA	User Name	MofnPubKey
1	PCO	admin	NO
0	NO		
2	AU	app_user	NO
0	NO		
3	CO	officer1	YES
0	NO		
4	CO	officer2	YES
0	NO		
5	CO	officer3	YES
0	NO		
6	CO	officer4	YES
0	NO		
7	CO	officer5	YES
0	NO		
8	CU	user1	NO
0	NO		

Se o CO tentar executar outra operação de gerenciamento de usuário HSM, ele falhará com um erro de autenticação de quorum, como mostrado no exemplo a seguir.

```

aws-cloudhsm > deleteUser CU user1
Deleting user user1(CU) on 2 nodes
deleteUser failed: RET_MXN_AUTH_FAILED
deleteUser failed on server 0(10.0.2.14)

Retry/rollBack/Ignore?(R/B/I): I
deleteUser failed: RET_MXN_AUTH_FAILED
deleteUser failed on server 1(10.0.1.4)

Retry/rollBack/Ignore?(R/B/I): I

```

O comando `listTokens` mostra que o CO não possui tokens aprovados, como mostrado no exemplo a seguir. Para realizar outra operação de gerenciamento de usuários do HSM, o CO deve obter um novo token de quorum, obter novas assinaturas dos aprovadores e aprovar o novo token no HSM.

```
aws-cloudhsm > listTokens
=====
      Server 0(10.0.2.14)
=====
Num of tokens = 0

=====
      Server 1(10.0.1.4)
=====
Num of tokens = 0

listTokens success
```

Altere o valor mínimo do quórum com AWS CloudHSM o Management Utility

Depois de [definir o valor mínimo do quórum](#) para que [os agentes AWS CloudHSM criptográficos \(COs\)](#) possam usar a autenticação de quórum, talvez você queira alterar o valor mínimo do quórum. O HSM permite que você altere o valor mínimo de quorum somente quando o número de aprovadores é igual ou superior ao valor mínimo do quorum atual. Por exemplo, se o valor mínimo do quórum for dois, pelo menos dois COs devem ser aprovados para alterar o valor mínimo do quórum.

Para obter a aprovação do quorum para alterar o valor mínimo do quorum, você precisa de um token de quorum para o comando `setMValue` (serviço 4). Para obter um token de quorum para o comando `setMValue` (serviço 4), o valor mínimo de quorum para o serviço 4 deve ser maior que um. Isso significa que antes de alterar o valor mínimo do quórum para COs (serviço 3), talvez seja necessário alterar o valor mínimo do quórum para o serviço 4.

A tabela a seguir lista os identificadores de serviço do HSM junto com seus nomes, descrições e comandos incluídos no serviço.

Identificadores de dispositivo	Nome do serviço	Descrição do serviço	Comandos do HSM
3	USER_MGMT	Gerenciamento de usuários do HSM	• <code>createUser</code> • <code>deleteUser</code>

Identificadores de dispositivo	Nome do serviço	Descrição do serviço	Comandos do HSM
			changePswd (aplica-se somente ao alterar a senha de outro usuário do HSM)
4	MISC_CO	Serviço de CO diverso	setMValue

Para alterar o valor mínimo do quorum para oficiais de criptografia

1. Use o comando a seguir para iniciar a ferramenta de linha de comando cloudhsm_mgmt_util.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

2. Use o comando loginHSM para fazer login nos HSMs como CO. Para obter mais informações, consulte [???](#).
3. Use o comando getMValue para obter o valor mínimo de quorum para o serviço 3. Para obter mais informações, veja o exemplo a seguir.
4. Use o comando getMValue para obter o valor mínimo de quorum para o serviço 4. Para obter mais informações, veja o exemplo a seguir.
5. Se o valor mínimo de quorum para o serviço 4 for menor que o valor para o serviço 3, use o comando setMValue para alterar o valor do serviço 4. Altere o valor do serviço 4 para um que seja igual ou superior ao valor do serviço 3. Para obter mais informações, veja o exemplo a seguir.
6. [Obtenha um token de quorum](#), tendo o cuidado de especificar o serviço 4 como o serviço para o qual você pode usar o token.
7. [Obtenha aprovações \(assinaturas\) de outros](#). COs
8. [Aprove o token no HSM](#).
9. Use o setMValue comando para alterar o valor mínimo do quórum para o serviço 3 (operações de gerenciamento de usuários realizadas por COs).

Example Obter valores mínimos de quorum e alterar o valor do serviço 4

O comando de exemplo a seguir mostra que o valor mínimo do quorum para o serviço 3 é atualmente dois.

```
aws-cloudhsm > getMValue 3
MValue of service 3[USER_MGMT] on server 0 : [2]
MValue of service 3[USER_MGMT] on server 1 : [2]
```

O comando de exemplo a seguir mostra que o valor mínimo de quorum para o serviço 4 é atualmente um.

```
aws-cloudhsm > getMValue 4
MValue of service 4[MISC_C0] on server 0 : [1]
MValue of service 4[MISC_C0] on server 1 : [1]
```

Para alterar o valor mínimo do quorum para o serviço 4, use o comando `setMValue`, definindo um valor igual ou superior ao valor do serviço 3. O exemplo a seguir define o valor mínimo do quorum para o serviço 4 como dois (2), o mesmo valor definido para o serviço 3.

```
aws-cloudhsm > setMValue 4 2
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)? y
Setting M Value(2) for 4 on 2 nodes
```

Os seguintes comandos mostram que o valor mínimo do quorum é agora dois para o serviço 3 e o serviço 4.

```
aws-cloudhsm > getMValue 3
MValue of service 3[USER_MGMT] on server 0 : [2]
MValue of service 3[USER_MGMT] on server 1 : [2]
```

```
aws-cloudhsm > getMValue 4
MValue of service 4[MISC_C0] on server 0 : [2]
```

```
MValue of service 4[MISC_C0] on server 1 : [2]
```

Entrada de chaves AWS CloudHSM

Antes de usar seu AWS CloudHSM cluster para processamento de criptografia, você deve criar [usuários](#) e chaves nos módulos de segurança de hardware (HSM) em seu cluster.

Em AWS CloudHSM, use qualquer um dos itens a seguir para gerenciar as chaves HSMs no seu cluster:

- Biblioteca PKCS #11
- Provedor JCE
- Provedores de KSP e CNG
- CLI do CloudHSM

Para gerenciar as chaves, faça login no HSM com o nome de usuário e a senha de um usuário de criptografia (CU). Somente um CU pode criar uma chave. O CU que cria uma chave possui e gerencia essa chave.

Consulte os tópicos a seguir para obter mais informações sobre como gerenciar chaves no AWS CloudHSM.

Tópicos

- [Configurações de sincronização e durabilidade de teclas em AWS CloudHSM](#)
- [Encapsulamento de chaves AES AWS CloudHSM](#)
- [Usando chaves confiáveis em AWS CloudHSM](#)
- [Gerenciamento de chaves com a CLI do CloudHSM](#)
- [Gerenciamento de chaves com o AWS CloudHSM KMU](#)

Configurações de sincronização e durabilidade de teclas em AWS CloudHSM

AWS CloudHSM sincroniza todas as chaves de token que você cria. A sincronização de chaves é basicamente um processo automático, mas você pode usar no mínimo dois módulos de segurança de hardware (HSM) em seu cluster para tornar as chaves mais duráveis. Este tópico descreve as principais configurações de sincronização, os problemas comuns que os clientes enfrentam ao trabalhar com chaves em um cluster e as estratégias para tornar as chaves mais duráveis.

Este tópico descreve as principais configurações de sincronização AWS CloudHSM, os problemas comuns que os clientes enfrentam ao trabalhar com chaves em um cluster e as estratégias para tornar as chaves mais duráveis.

Tópicos

- [AWS CloudHSM conceitos-chave](#)
- [Entendendo AWS CloudHSM a sincronização de teclas](#)
- [Alterar as configurações de durabilidade da chave do AWS CloudHSM cliente](#)
- [Sincronização de chaves em clusters clonados AWS CloudHSM](#)

AWS CloudHSM conceitos-chave

Os conceitos necessários para trabalhar com chaves no AWS CloudHSM são os mostrados a seguir.

Chaves de token

Chaves persistentes que você cria durante as operações de geração, importação ou desempacotamento de chaves. AWS CloudHSM sincroniza chaves de token em um cluster.

Chaves de sessão

Chaves efêmeras que existem somente em um módulo de segurança de hardware (HSM) no cluster. AWS CloudHSM não sincroniza as chaves de sessão em um cluster.

Sincronização de chaves do lado do cliente

Um processo do lado do cliente que clona as chaves de token que você cria durante as operações de geração, importação ou desencapsulamento de chaves. Você pode tornar as chaves de token mais duráveis executando um cluster com no mínimo duas HSMs.

Sincronização de chaves do lado do servidor

Clona periodicamente as chaves de cada HSM no cluster. Não requer gerenciamento.

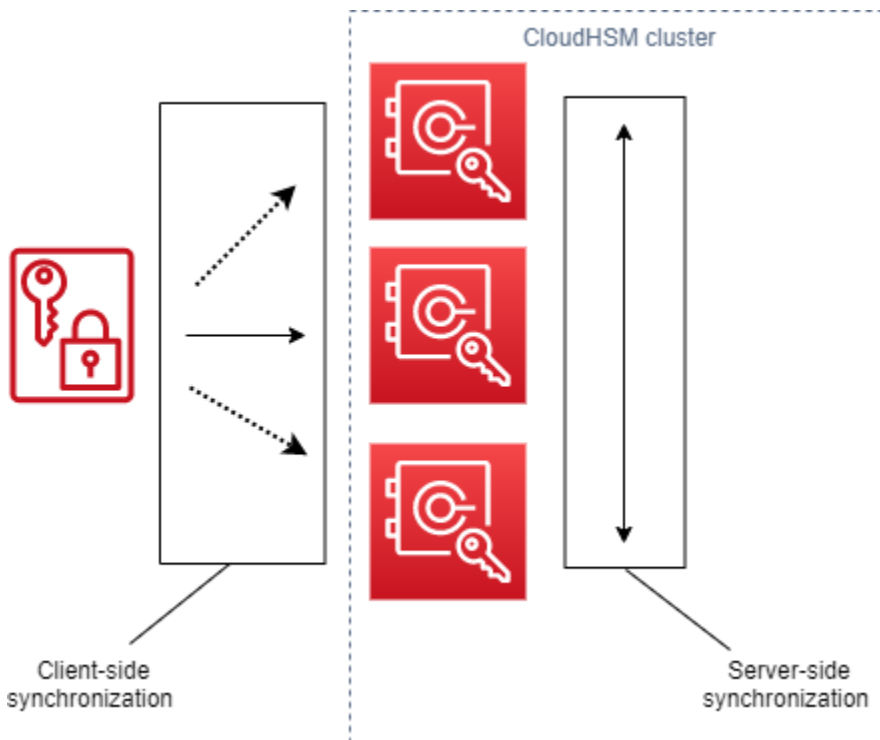
Configurações de durabilidade de chave do cliente

Configurações que você define no cliente que afetam a durabilidade de chave. Essas configurações funcionam de forma diferente no Client SDK 5 e no Client SDK 3.

- No Client SDK 5, use essa configuração para executar um único cluster HSM.
- No SDK do cliente 3, use essa configuração para especificar o número HSMs necessário para que as operações de criação de chaves sejam bem-sucedidas.

Entendendo AWS CloudHSM a sincronização de teclas

AWS CloudHSM usa a sincronização de chaves para clonar chaves de token em todos os módulos de segurança de hardware (HSM) em um cluster. Você cria chaves de token como chaves persistentes durante as operações de geração, importação ou desencapsulamento de chaves. Para distribuir essas chaves em todo o cluster, o CloudHSM oferece a sincronização de chaves do lado do cliente e do lado do servidor.



O objetivo da sincronização de chaves, tanto do lado do servidor quanto do lado do cliente, é distribuir as novas chaves pelo cluster o mais rápido possível depois de criá-las. Isso é importante porque as chamadas subsequentes que você fizer para usar novas chaves podem ser roteadas para qualquer HSM disponível no cluster. Se a chamada que você fizer for encaminhada para um HSM sem a chave, a chamada falhará. Você pode mitigar esses tipos de falhas especificando que seus aplicativos tentem novamente as chamadas subsequentes feitas após as operações de criação da chave. O tempo necessário para a sincronização pode variar, dependendo da workload do seu cluster e de outros intangíveis. Use CloudWatch métricas para determinar o tempo que seu aplicativo deve empregar nesse tipo de situação. Para obter mais informações, consulte [CloudWatch Métricas do](#).

O desafio da sincronização de chaves em um ambiente de nuvem é a durabilidade delas. Você cria chaves em um único HSM e geralmente começa a usá-las imediatamente. Se o HSM no qual você criou as chaves falhar antes delas serem clonadas em outro HSM no cluster, você perderá as

chaves e o acesso a qualquer coisa criptografada por elas. Para mitigar esse risco, oferecemos a sincronização do lado do cliente. Esta sincronização é um processo do lado do cliente que clona as chaves que você cria durante as operações de geração, importação ou desencapsulamento de chaves. A clonagem das chaves à medida que você as cria torna as chaves mais duráveis. Obviamente, você não pode clonar chaves em um cluster com um único HSM. Para tornar as chaves mais duráveis, também recomendamos que você configure seu cluster para usar no mínimo duas HSMs. Com a sincronização do lado do cliente e um cluster com dois HSMs, você pode enfrentar o desafio da durabilidade das chaves em um ambiente de nuvem.

Alterar as configurações de durabilidade da chave do AWS CloudHSM cliente

A sincronização de chaves é basicamente um processo automático, mas você pode gerenciar as configurações de durabilidade de chaves do lado do cliente. As configurações de durabilidade de chave do lado do cliente funcionam de forma diferente no Client SDK 5 e no Client SDK 3.

- No Client SDK 5, apresentamos o conceito de quóruns de disponibilidade de chaves, que exige que você execute clusters com no mínimo dois HSMs. Você pode usar as configurações de durabilidade de chave do lado do cliente para optar por não atender aos dois requisitos de HSM. Para obter mais informações sobre quoruns, consulte [the section called “Conceitos do Client SDK 5”](#).
- No Client SDK 3, você usa as configurações de durabilidade da chave do lado do cliente para especificar o número HSMs em que a criação da chave deve ser bem-sucedida para que a operação geral seja considerada bem-sucedida.

Configurações de durabilidade de chave do Client SDK 5

No Client SDK 5, a sincronização de chaves é um processo totalmente automático. Com o quórum de disponibilidade de chaves, as chaves recém-criadas devem existir em duas HSMs no cluster antes que seu aplicativo possa usar a chave. Para usar o quórum de disponibilidade de chaves, seu cluster deve ter no mínimo dois HSMs.

Se a configuração do cluster não atender aos principais requisitos de durabilidade, qualquer tentativa de criar ou usar uma chave de token falhará com a seguinte mensagem de erro nos registros:

```
Key <key handle> does not meet the availability requirements - The key must be available on at least 2 HSMs before being used.
```

Você pode usar as configurações do cliente para cancelar o quórum de disponibilidade de chaves. Talvez você queira optar por não executar um cluster com um único HSM, por exemplo.

Conceitos do Client SDK 5

Quórum de disponibilidade chave

AWS CloudHSM especifica o número de HSMs em um cluster no qual as chaves devem existir antes que seu aplicativo possa usar a chave. Requer clusters com no mínimo dois HSMs.

Gerenciando as configurações de durabilidade de chave do cliente

Para gerenciar as configurações de durabilidade de chave do cliente, você deve usar a ferramenta de configuração do Client SDK 5.

PKCS #11 library

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --disable-key-availability-check
```

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --disable-key-availability-check
```

OpenSSL Dynamic Engine

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.


```
$ sudo /opt/cloudhsm/bin/configure-dyn --disable-key-availability-check
```

OpenSSL Dynamic Engine Provider

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --disable-key-availability-check
```

Key Storage Provider (KSP)

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --disable-key-availability-check
```

JCE provider

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-jce --disable-key-availability-check
```

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --disable-key-availability-check
```

CloudHSM CLI

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-cli --disable-key-availability-check
```

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --disable-key-availability-check
```

Configurações de durabilidade de chave do Client SDK 3

No Client SDK 3, a sincronização de chaves é basicamente um processo automático, mas você pode usar as configurações de durabilidade de chave do cliente para tornar as chaves mais duráveis. Você especifica o número HSMs em que a criação da chave deve ser bem-sucedida para que a operação geral seja considerada bem-sucedida. A sincronização do lado do cliente sempre faz o possível para clonar as chaves de cada HSM no cluster, independentemente da configuração escolhida. Sua configuração impõe a criação de chaves no número HSMs que você especificar. Se você especificar um valor e o sistema não conseguir replicar a chave para esse número de HSMs, o sistema limpará automaticamente qualquer material de chave indesejado e você poderá tentar novamente.

Important

Se você não definir as configurações de durabilidade da chave do cliente (ou se usar o valor padrão de 1), suas chaves estarão vulneráveis à perda. Se seu HSM atual falhar antes que

o serviço do lado do servidor tenha clonado essa chave em outro HSM, você perderá o material da chave.

Para maximizar a durabilidade da chave, considere especificar pelo menos duas HSMs para sincronização do lado do cliente. Lembre-se de que não importa quantas HSMs você especifique, a carga de trabalho em seu cluster permanece a mesma. A sincronização do lado do cliente sempre faz o possível para clonar as chaves de cada HSM no cluster.

Recomendações

- Mínimo: dois HSMs por cluster
- Máximo: um a menos do que o número total de HSMs em seu cluster

Se a sincronização do lado do cliente falhar, o serviço do cliente limpará todas as chaves indesejadas que possam ter sido criadas e que agora sejam indesejadas. Essa limpeza é a melhor resposta que nem sempre pode funcionar. Se a limpeza falhar, talvez seja necessário excluir o material de chave indesejado. Para obter mais informações, consulte [Falhas de sincronização de chaves](#).

Configurando o arquivo de configuração para durabilidade de chave do cliente

Para especificar as configurações de durabilidade de chave do cliente, você deve editar `cloudhsm_client.cfg`.

Para editar a configuração do cliente

1. Abra o `cloudhsm_client.cfg`.

Linux

```
/opt/cloudhsm/etc/cloudhsm_client.cfg
```

Windows:

```
C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

2. No `client` nó do arquivo, adicione `create_object_minimum_nodes` e especifique um valor para o número mínimo HSMs em que as chaves AWS CloudHSM devem ser criadas com êxito para que as operações de criação de chaves sejam bem-sucedidas.

```
"create_object_minimum_nodes" : 2
```

Note

A ferramenta de linha de comando `key_mgmt_util` (KMU) tem uma configuração adicional para durabilidade da chave do cliente. Para obter mais informações, consulte [the section called “KMU e sincronização do lado do cliente”](#).

Referência da configuração

Essas são as propriedades de sincronização do lado do cliente, mostradas em um trecho do `cloudhsm_client.cfg`:

```
{
  "client": {
    "create_object_minimum_nodes" : 2,
    ...
  },
  ...
}
```

`create_object_minimum_nodes`

Especifica o número mínimo HSMs necessário para considerar as operações de geração, importação ou descompactamento de chaves bem-sucedidas. O padrão é definido como "1". Isso significa que, para cada operação de criação de chave, o serviço do lado do cliente tenta criar chaves em cada HSM no cluster, mas, para retornar um sucesso, só é preciso criar uma única chave em um HSM no cluster.

KMU e sincronização do lado do cliente

Se você criar chaves com a ferramenta de linha de comando `key_mgmt_util` (KMU), você usa um parâmetro de linha de comando opcional (`-min_srv`) para limitar o número de chaves nas quais

clonar. HSMs Se você especificar o parâmetro da linha de comando e um valor no arquivo de configuração, AWS CloudHSM honra o MAIOR dos dois valores.

Para saber mais, consulte os seguintes tópicos:

- [2 gênios DSAKey Pair](#)
- [2 gênios ECCKey Pair](#)
- [2 gênios RSAKey Pair](#)
- [genSymKey](#)
- [importPrivateKey](#)
- [importPubKey](#)
- [imSymKey](#)
- [insertMaskedObject](#)
- [unWrapKey](#)

Sincronização de chaves em clusters clonados AWS CloudHSM

A sincronização do lado do cliente e do lado do servidor serve apenas para sincronizar chaves dentro do mesmo cluster. AWS CloudHSM Se você copiar um backup de um cluster para outra região, use o comando [key replicate](#) para replicar uma chave entre dois clusters. Você pode usar clusters clonados para redundância entre regiões ou para simplificar seu processo de recuperação de desastres. Se não tiver instalado a CloudHSM CLI, siga as instruções em [???](#).

Encapsulamento de chaves AES AWS CloudHSM

Este tópico descreve as opções de empacotamento de chaves AES. AWS CloudHSM O encapsulamento de chaves AES usa uma chave AES (a chave de encapsulamento) para encapsular outra chave de qualquer tipo (a chave de destino). O encapsulamento de chaves é usado para proteger chaves armazenadas ou para transmitir chaves por redes não seguras.

Tópicos

- [Algoritmos compatíveis](#)
- [Usando o encapsulamento de teclas AES AWS CloudHSM](#)

Algoritmos compatíveis

AWS CloudHSM oferece três opções para empacotamento de chaves AES, cada uma com base em como a chave de destino é preenchida antes de ser embalada. O preenchimento é feito automaticamente, de acordo com o algoritmo usado, quando você chama o encapsulamento de chaves. A tabela a seguir lista os algoritmos compatíveis e os detalhes associados para ajudar você a escolher um mecanismo de encapsulamento apropriado para seu aplicativo.

Algoritmo de encapsulamento de chave AES	Especificação	Tipos de chave de destino compatíveis	Esquema de preenchimento	AWS CloudHSM Disponibilidade do cliente
Encapsulamento de chaves AES com preenchimento de zeros	RFC 5649 e SP 800-38F	Todos	Adiciona zeros após bits de chave, se necessário, para bloquear o alinhamento	SDK 3.1 e posterior
Encapsulamento de chaves AES sem preenchimento	RFC 3394 e SP 800-38F	Chaves com alinhamento bloqueado, como AES e 3DES	Nenhum	SDK 3.1 e posterior
Encapsulamento de chaves AES com preenchimento PKCS #5	Nenhum	Todos	Pelo menos 8 bytes são adicionados de acordo com o esquema de preenchimento PKCS #5 para bloquear o alinhamento	Todos

Para saber como usar os algoritmos de encapsulamento de chaves AES da tabela anterior no aplicativo, consulte [Usar o encapsulamento de chaves AES no AWS CloudHSM](#).

Noções básicas sobre vetores de inicialização no encapsulamento de chaves AES

Antes de encapsular, o CloudHSM anexa um vetor de inicialização (IV) à chave de destino para a integridade dos dados. Cada algoritmo de encapsulamento de chaves tem restrições específicas sobre o tipo de IV permitido. Para configurar o IV AWS CloudHSM, você tem duas opções:

- Implícito: defina o IV como NULL e o CloudHSM usará o valor padrão desse algoritmo para operações de encapsulamento e desencapsulamento (recomendado)
- Explícito: defina o IV transmitindo o valor padrão do IV para a função de encapsulamento de chaves

Important

É necessário entender o IV que você está usando no aplicativo. Para desencapsular a chave, é necessário fornecer o mesmo IV usado para encapsular a chave. Se você usar um IV implícito para encapsular, use um IV implícito para desencapsular. Com um IV implícito, o CloudHSM usará o valor padrão para desencapsular.

A tabela a seguir descreve os valores permitidos para os quais IVs o algoritmo de empacotamento especifica.

Algoritmo de encapsulamento de chave AES	IV implícito	IV explícito
Encapsulamento de chaves AES com preenchimento de zeros	Obrigatório Valor padrão: (IV calculado internamente com base na especificação)	Não permitido
Encapsulamento de chaves AES sem preenchimento	Permitido (recomendado) Valor padrão: 0xA6A6A6A6A6A6A6A6	Permitido Apenas este valor aceito: 0xA6A6A6A6A6A6A6A6

Algoritmo de encapsulamento de chave AES	IV implícito	IV explícito
Encapsulamento de chaves AES com preenchimento PKCS #5	Permitido (recomendado) Valor padrão: 0xA6A6A6A6A6A6A6A6	Permitido Apenas este valor aceito: 0xA6A6A6A6A6A6A6A6

Usando o encapsulamento de teclas AES AWS CloudHSM

Encapsule e desencapsule as chaves da seguinte maneira:

- Na [biblioteca PCKS #11](#), selecione o mecanismo apropriado para as funções `C_WrapKey` e `C_UnWrapKey`, conforme mostrado na tabela a seguir.
- Na [provedor JCE](#), selecione a combinação apropriada de algoritmo, modo e preenchimento, implementando os métodos de criptografia `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`, conforme mostrado na tabela a seguir.
- Na [CLI do CloudHSM](#), escolha o algoritmo apropriado na lista de algoritmos compatíveis em [O comando key wrap na CLI do CloudHSM](#) e [O comando key unwrap na CLI do CloudHSM](#), conforme mostrado na tabela a seguir.
- Em [key_mgmt_util \(KMU\)](#), use os comandos [Exportar uma AWS CloudHSM chave usando o KMU](#) e [Desembrulhe uma AWS CloudHSM chave usando KMU](#) com valores `m` apropriados, conforme mostrado na tabela a seguir.

Algoritmo de encapsulamento de chave AES	Mecanismo PKCS #11	Método Java	Subcomando da CLI do CloudHSM	Argumento do Key Management Utility (KMU)
Encapsulamento de chaves AES com preenchimento de zeros	CKM_CLOUD_HSM_AES_KEY_WRAP_ZERO_PAD (mecanismo definido pelo fornecedor)	AESWrap/ECB/ZeroPadding	aes-zero-pad	<code>m = 6</code>

Algoritmo de encapsulamento de chave AES	Mecanismo PKCS #11	Método Java	Subcomando da CLI do CloudHSM	Argumento do Key Management Utility (KMU)
Encapsulamento de chaves AES sem preenchimento	CKM_CLOUD_HSM_AES_KEY_WRAP_NO_PAD (mecanismo definido pelo fornecedor)	AESWrap/ECB/NoPadding	aes-no-pad	m = 5
Encapsulamento de chaves AES com preenchimento PKCS #5	CKM_CLOUD_HSM_AES_KEY_WRAP_PKCS5_PAD (mecanismo definido pelo fornecedor)	AESWrap/ECB/PKCS5Padding	aes-pkcs5-pad	m = 4

Usando chaves confiáveis em AWS CloudHSM

AWS CloudHSM oferece suporte ao agrupamento confiável de chaves para proteger as chaves de dados contra ameaças internas. Este tópico descreve como criar chaves confiáveis para proteger os dados.

Tópicos

- [Entendendo as chaves confiáveis em AWS CloudHSM](#)
- [Atributos-chave confiáveis em AWS CloudHSM](#)
- [Como usar chaves confiáveis para agrupar chaves de dados AWS CloudHSM](#)
- [Como desempacotar uma chave de dados com uma chave confiável para AWS CloudHSM](#)

Entendendo as chaves confiáveis em AWS CloudHSM

Uma chave confiável é uma chave usada para agrupar outras chaves e que administradores e agentes criptográficos (COs) identificam especificamente como confiáveis usando o atributo `CKA_TRUSTED`. Além disso, administradores e agentes criptográficos (COs) usam `CKA_UNWRAP_TEMPLATE` atributos relacionados para especificar quais ações as chaves de dados podem realizar quando são desempacotadas por uma chave confiável. As chaves de dados desagrupadas pela chave confiável também devem conter esses atributos para que a operação de desagrupamento seja bem-sucedida, o que ajuda a garantir que as chaves de dados desagrupadas sejam permitidas somente para o uso pretendido.

Use o atributo `CKA_WRAP_WITH_TRUSTED` para identificar todas as chaves de dados que você deseja agrupar com chaves confiáveis. Isso permite restringir as chaves de dados para que os aplicativos só possam usar chaves confiáveis para desagrupá-las. Depois de definir esse atributo nas chaves de dados, o atributo se tornará somente leitura e não será possível alterá-lo. Com esses atributos em vigor, os aplicativos só podem desagrupar suas chaves de dados com as chaves em que você confia, e desagrupamentos sempre resultam em chaves de dados com atributos que limitam a forma como essas chaves podem ser usadas.

Atributos-chave confiáveis em AWS CloudHSM

Os atributos a seguir permitem marcar uma AWS CloudHSM chave como confiável, especificar que uma chave de dados só pode ser encapsulada e desempacotada com uma chave confiável e controlar o que uma chave de dados pode fazer depois de desempacotada:

- **CKA_TRUSTED:** aplique esse atributo (além de `CKA_UNWRAP_TEMPLATE`) à chave que agrupará as chaves de dados para especificar que um administrador ou responsável pela criptografia (CO) fez a diligência necessária e confia nessa chave. Somente um administrador ou CO pode configurar um `CKA_TRUSTED`. O usuário de criptografia (UC) possui a chave, mas somente um CO pode definir o atributo `CKA_TRUSTED`.
- **CKA_WRAP_WITH_TRUSTED:** Aplique esse atributo a uma chave de dados exportável para especificar que você só pode agrupar essa chave com chaves marcadas como `CKA_TRUSTED`. Depois que `CKA_WRAP_WITH_TRUSTED` for definido como verdadeiro, o atributo se torna somente para leitura e não é possível alterar ou remover o atributo.
- **CKA_UNWRAP_TEMPLATE:** aplique esse atributo à chave de agrupamento (além de `CKA_TRUSTED`) para especificar quais nomes e valores de atributos o serviço deve aplicar automaticamente às chaves de dados que o serviço desagrupa. Quando um aplicativo envia uma chave para

desencapsulamento, ele pode fornecer separadamente um modelo de desencapsulamento. Se você especificar um modelo de desagrupamento e o aplicativo fornecer seu próprio modelo de desagrupamento, o HSM usará os dois modelos para aplicar nomes e valores de atributos à chave. No entanto, se um valor no CKA_UNWRAP_TEMPLATE para a chave de encapsulamento entrar em conflito com um atributo fornecido pelo aplicativo durante a solicitação de desencapsulamento, ocorrerá uma falha na solicitação de desencapsulamento.

Para obter mais informações, consulte os tópicos a seguir:

- [Atributos de chaves do PKCS #11](#)
- [Atributos de chaves do JCE](#)
- [Atributos de chaves da CLI do CloudHSM](#)

Como usar chaves confiáveis para agrupar chaves de dados AWS CloudHSM

Para usar uma chave confiável para agrupar uma chave de dados AWS CloudHSM, você deve concluir três etapas básicas:

1. Para a chave de dados que você planeja agrupar com uma chave confiável, defina seu atributo CKA_WRAP_WITH_TRUSTED como verdadeiro.
2. Para a chave confiável com a qual você planeja agrupar a chave de dados, defina seu atributo CKA_TRUSTED como verdadeiro.
3. Use a chave confiável para agrupar a chave de dados.

Etapa 1: defina a chave de dados **CKA_WRAP_WITH_TRUSTED** como verdadeira

Para a chave de dados que você deseja agrupar, escolha uma das opções a seguir para definir o atributo CKA_WRAP_WITH_TRUSTED da chave como verdadeiro. Isso restringe a chave de dados para que os aplicativos possam usar apenas chaves confiáveis para agrupá-las.

Opção 1: se estiver gerando uma nova chave, defina **CKA_WRAP_WITH_TRUSTED** como verdadeira

Gere uma chave usando [PKCS #11](#), [JCE](#) ou [CloudHSM CLI](#). Consulte o código a seguir para ver um exemplo.

PKCS #11

Para gerar uma chave com PKCS #11, você precisa definir o atributo CKA_WRAP_WITH_TRUSTED da chave como verdadeiro. Conforme mostrado no exemplo a seguir, faça isso incluindo este atributo no CK_ATTRIBUTE template da chave e, em seguida, definindo o atributo como verdadeiro:

```
CK_BYTE_PTR label = "test_key";
CK_ATTRIBUTE template[] = {
    {CKA_WRAP_WITH_TRUSTED, &true_val,      sizeof(CK_BBOOL)},
    {CKA_LABEL,             label,           strlen(label)},
    ...
};
```

Para obter mais informações, consulte [nossos exemplos públicos que demonstram a geração de chaves com o PKCS #11](#).

JCE

Para gerar uma chave com JCE, você precisa definir o atributo WRAP_WITH_TRUSTED da chave como verdadeiro. Conforme mostrado no exemplo a seguir, faça isso incluindo este atributo no KeyAttributesMap da chave e, em seguida, definindo o atributo como verdadeiro:

```
final String label = "test_key";
final KeyAttributesMap keySpec = new KeyAttributesMap();
keySpec.put(KeyAttribute.WRAP_WITH_TRUSTED, true);
keySpec.put(KeyAttribute.LABEL, label);
...
```

Para obter mais informações, consulte [nossos exemplos públicos que demonstram a geração de chaves com JCE](#).

CloudHSM CLI

Para gerar uma chave com a CLI do CloudHSM, você precisa definir o atributo wrap-with-trusted da chave como verdadeiro. Faça isso incluindo wrap-with-trusted=true no argumento apropriado para o comando de geração de chave:

- Para chaves simétricas, adicione wrap-with-trusted ao argumento attributes.
- Para chaves públicas, adicione wrap-with-trusted ao argumento public-attributes.

- Para chaves privadas, adicione `wrap-with-trusted` ao argumento `private-attributes`.

Para obter mais informações sobre pares de chaves, consulte [A generate-asymmetric-pair categoria na CLI do CloudHSM](#).

Para obter mais informações sobre pares de chaves simétricas, consulte [A categoria generate-symmetric na CLI do CloudHSM](#).

Opção 2: se estiver usando uma chave existente, use a CLI do CloudHSM para defini-la como verdadeira **CKA_WRAP_WITH_TRUSTED**

Para definir o atributo `CKA_WRAP_WITH_TRUSTED` de uma chave existente como verdadeiro, siga estas etapas:

1. Use o comando [Fazer login em um HSM usando a CLI do CloudHSM](#) para fazer login como usuário de criptografia (CU).
2. Use o comando [Definir os atributos das chaves com a CloudHSM CLI](#) para definir o atributo da chave `wrap-with-trusted` como verdadeiro.

```
aws-cloudhsm > key set-attribute --filter attr.label=test_key --name wrap-with-trusted --value true
{
  "error_code": 0,
  "data": {
    "message": "Attribute set successfully"
  }
}
```

Etapa 2: defina as chaves confiáveis **CKA_TRUSTED** como verdadeiras

Para tornar uma chave confiável, o atributo `CKA_TRUSTED` deve ser definido como verdadeiro. É possível usar a CLI do CloudHSM ou o CloudHSM Management Utility (CMU) para fazer isso.

- Se estiver usando a CLI do CloudHSM para definir o atributo de uma chave `CKA_TRUSTED`, consulte [Marcar uma chave como confiável usando a CLI do CloudHSM](#)
- Se estiver usando o CMU para definir o atributo `CKA_TRUSTED` de uma chave, consulte [Como marcar uma chave como confiável com o Utilitário AWS CloudHSM de Gerenciamento](#).

Etapa 3. Use a chave confiável para agrupar a chave de dados

Para agrupar a chave de dados mencionada na Etapa 1 com a chave confiável que você definiu na Etapa 2, consulte os links a seguir para ver exemplos de código. Cada um demonstra como agrupar as chaves.

- [AWS CloudHSM Exemplos de PKCS #11](#)
- [AWS CloudHSM Exemplos de JCE](#)

Como desempacotar uma chave de dados com uma chave confiável para AWS CloudHSM

Para desempacotar uma chave de dados AWS CloudHSM, você precisa de uma chave confiável que tenha sido CKA_UNWRAP definida como verdadeira. Para ser uma chave desse tipo, ela também deve atender aos seguintes critérios:

- O atributo CKA_TRUSTED da chave deve ser definido como verdadeiro.
- A chave deve usar atributos CKA_UNWRAP_TEMPLATE relacionados para especificar quais ações as chaves de dados podem realizar depois de desagrupadas. Se, por exemplo, você quiser que uma chave desagrupada não seja exportável, defina CKA_EXPORTABLE = FALSE como parte do CKA_UNWRAP_TEMPLATE.

Note

CKA_UNWRAP_TEMPLATE só está disponível com o PKCS #11.

Quando um aplicativo envia uma chave para ser desagrupada, ele pode fornecer separadamente um modelo de desagrupamento. Se você especificar um modelo de desagrupamento e o aplicativo fornecer seu próprio modelo de desagrupamento, o HSM usará os dois modelos para aplicar nomes e valores de atributos à chave. No entanto, se durante a solicitação de desagrupamento um valor na chave confiável entrar em CKA_UNWRAP_TEMPLATE conflito com um atributo fornecido pelo aplicativo, a solicitação de desagrupamento falhará.

Para ver um exemplo de como desagrupar uma chave de dados com uma chave confiável, consulte [este exemplo de PKCS #11](#).

Gerenciamento de chaves com a CLI do CloudHSM

Se estiver usando a [série de versões mais recente do SDK](#), use a [CLI do CloudHSM](#) para gerenciar as chaves em seu cluster. AWS CloudHSM Para obter mais informações, consulte os tópicos abaixo.

- [O uso de chaves confiáveis](#) descreve como usar a CLI do CloudHSM para criar chaves confiáveis para proteger os dados.
- A [geração de chaves](#) inclui instruções sobre a criação de chaves, incluindo chaves simétricas, chaves RSA e chaves EC.
- A [exclusão de chaves](#) descreve como os proprietários das chaves excluem as chaves.
- O [compartilhamento e o não compartilhamento de chaves](#) detalham como os proprietários compartilham e cancelam o compartilhamento das chaves.
- A [filtragem de chaves](#) oferece diretrizes sobre como usar filtros para encontrar chaves.
- [Gerenciar a autenticação de quórum com chaves \(M de N\)](#) oferece diretrizes sobre como configurar e usar a autenticação de quórum com chaves.

Gerar chaves com a CLI do CloudHSM

Antes de gerar uma chave, você deve iniciar a [CLI do CloudHSM](#) e fazer login como um usuário de criptografia (CU). Para gerar chaves no HSM, use o comando que corresponde ao tipo de chave que você deseja gerar.

Tópicos

- [Gerar chaves simétricas com a CLI do CloudHSM](#)
- [Gerar chaves assimétricas usando a CLI do CloudHSM](#)
- [AWS CloudHSM principais tópicos relacionados](#)

Gerar chaves simétricas com a CLI do CloudHSM

Use os comandos listados em [A categoria generate-symmetric na CLI do CloudHSM](#) para gerar chaves simétricas para AWS CloudHSM. Para consultar todas as opções disponíveis, use o comando `help key generate-symmetric`.

Gerar uma chave de AES.

Use o comando `key generate-symmetric aes` para gerar chaves AES. Para consultar todas as opções disponíveis, use o comando `help key generate-symmetric aes`.

Example

O exemplo a seguir gera uma chave AES de 32 bytes.

```
aws-cloudhsm > key generate-symmetric aes \  
  --label aes-example \  
  --key-length-bytes 32
```

Argumentos

<LABEL>

Especifica o rótulo da chave privada definida pelo usuário para a chave AES.

Obrigatório: Sim

<KEY-LENGTH-BYTES>

Especifica o tamanho da chave em bytes.

Valores válidos:

- 16, 24 e 32

Obrigatório: Sim

<KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave AES gerada na forma de `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (por exemplo, `sign=true`)

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão. Use esse parâmetro quando precisar de uma chave apenas brevemente,

como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [key set-attribute](#).

Por padrão, quando as chaves são geradas, elas são persistent/token chaves. Usar <SESSION> muda isso, garantindo que uma chave gerada com esse argumento seja uma sessão/efêmera

Obrigatório: não

Gerar chave secreta genérica

Use o comando `key generate-symmetric generic-secret` para gerar chaves secretas genéricas. Para consultar todas as opções disponíveis, use o comando `help key generate-symmetric generic-secret`.

Example

O exemplo a seguir gera uma chave secreta genérica de 32 bytes.

```
aws-cloudhsm > key generate-symmetric generic-secret \  
  --label generic-secret-example \  
  --key-length-bytes 32
```

Argumentos

<LABEL>

Especifica um rótulo definido pelo usuário para a chave secreta genérica.

Obrigatório: Sim

<KEY-LENGTH-BYTES>

Especifica o tamanho da chave em bytes.

Valores válidos:

- 1 a 800

Obrigatório: Sim

<KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave secreta genérica gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true)

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão. Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [key set-attribute](#).

Por padrão, quando as chaves são geradas, elas são persistent/token chaves. Usar <SESSION> muda isso, garantindo que uma chave gerada com esse argumento seja uma sessão/efêmera

Obrigatório: não

Gerar chaves assimétricas usando a CLI do CloudHSM

Use os comandos listados em [A generate-asymmetric-pair categoria na CLI do CloudHSM](#) para gerar pares de chaves assimétricas para AWS CloudHSM clusters.

Gerar uma chave RSA

Use o comando `key generate-asymmetric-pair rsa` para gerar um par de chaves RSA. Para consultar todas as opções disponíveis, use o comando `help key generate-asymmetric-pair rsa`.

Example

O exemplo a seguir gera um par de chaves RSA de 2048 bits.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \  
  --public-exponent 65537 \  
  --modulus-size-bits 2048 \  
  --key-attributes sign=true
```

```
--public-label rsa-public-example \  
--private-label rsa-private-example
```

Argumentos

<PUBLIC_LABEL>

Especifica um rótulo definido pelo usuário para a chave pública.

Obrigatório: Sim

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário.

Obrigatório: Sim

<MODULUS_SIZE_BITS>

Especifica o comprimento do módulo em bits. O valor mínimo é 2048.

Obrigatório: Sim

<PUBLIC_EXPONENT>

Especifica o expoente público. O valor deve ser um número ímpar maior que ou igual a 65537.

Obrigatório: Sim

<PUBLIC_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave pública RSA na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true).

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão. Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [key set-attribute](#).

Por padrão, quando as chaves são geradas, elas são persistent/token chaves. Usar <SESSION> muda isso, garantindo que uma chave gerada com esse argumento seja uma sessão/efêmera

Obrigatório: não

Gerar pares de chaves EC (protocolo de criptografia de curva elíptica)

Use o comando `key generate-asymmetric-pair ec` para gerar um par de chaves EC. Para consultar todas as opções disponíveis, incluindo uma lista das curvas elípticas suportadas, use o comando `help key generate-asymmetric-pair ec`.

Example

O exemplo a seguir gera um par de chaves ECC usando a curva elíptica Secp384r1.

```
aws-cloudhsm > key generate-asymmetric-pair ec \  
  --curve secp384r1 \  
  --public-label ec-public-example \  
  --private-label ec-private-example
```

Argumentos

<PUBLIC_LABEL>

Especifica um rótulo definido pelo usuário para a chave pública. O tamanho máximo permitido para `label` é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário. O tamanho máximo permitido para `label` é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<CURVE>

Especifica o identificador da curva elíptica.

Valores válidos:

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1
- ed25519 (compatível somente em instâncias hsm2m.medium no modo não FIPS)

Obrigatório: Sim

<PUBLIC_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos principais atributos a serem definidos para a chave pública EC gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, verify=true).

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<PRIVATE_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos principais atributos a serem definidos para a chave privada EC gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true).

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão. Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [key set-attribute](#).

Por padrão, as chaves geradas são chaves persistentes (tokens). Passar <SESSION> muda isso, garantindo que uma chave gerada com esse argumento seja uma chave de sessão (efêmera).

Obrigatório: não

AWS CloudHSM principais tópicos relacionados

Consulte as seções a seguir para obter mais informações sobre as chaves de entrada no AWS CloudHSM.

- [Atributos de chave da CloudHSM CLI](#)
- [A generate-asymmetric-pair categoria na CLI do CloudHSM](#)
- [A categoria generate-symmetric na CLI do CloudHSM](#)

Excluir chaves usando a CLI do CloudHSM

Use o exemplo neste tópico para excluir uma chave com a CLI do [CloudHSM](#). Somente os proprietários da chave podem excluí-la.

Tópicos

- [Exemplo: excluir uma chave](#)
- [Tópicos relacionados](#)

Exemplo: excluir uma chave

1. Execute o comando `key list` para identificar a chave que você quer excluir:

```
aws-cloudhsm > key list --filter attr.label="my_key_to_delete" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000540011",
        "key-info": {
          "key-owners": [
            {
              "username": "my_crypto_user",
```

```

        "key-coverage": "full"
    }
],
"shared-users": [],
"key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
},
"cluster-coverage": "full"
},
"attributes": {
    "key-type": "rsa",
    "label": "my_key_to_delete",
    "id": "",
    "check-value": "0x29bbd1",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1217,
    "public-exponent": "0x010001",
    "modulus":
"0x8b3a7c20618e8be08220ed8ab2c8550b65fc1aad8d4cf04fbf2be685f97eeb78fcbbad9b02cd91a3b15e990
    "modulus-size-bits": 2048
}
}
],
"total_key_count": 1,
"returned_key_count": 1

```

```
}
```

2. Depois de identificar a chave, execute o key delete com o atributo exclusivo label da chave para excluí-la:

```
aws-cloudhsm > key delete --filter attr.label="my_key_to_delete"
{
  "error_code": 0,
  "data": {
    "message": "Key deleted successfully"
  }
}
```

3. Execute o comando key list com o atributo exclusivo label da chave e confirme se a chave foi excluída. Conforme mostrado no exemplo a seguir, nenhuma chave com o rótulo my_key_to_delete está no cluster do HSM:

```
aws-cloudhsm > key list --filter attr.label="my_key_to_delete"
{
  "error_code": 0,
  "data": {
    "matched_keys": [],
    "total_key_count": 0,
    "returned_key_count": 0
  }
}
```

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Excluir uma chave com a CLI do CloudHSM](#)

Compartilhar e cancelar o compartilhamento de chaves usando a CLI do CloudHSM

Use os comandos neste tópico para compartilhar e cancelar o compartilhamento de chaves na [CLI do CloudHSM](#). Em AWS CloudHSM, o usuário criptográfico (UC) que cria a chave é o proprietário dela. O proprietário pode usar os key unshare comandos key share e para compartilhar e descompartilhar a chave com outras CUs pessoas. Os usuários com quem a chave é compartilhada

podem usá-la em operações criptográficas, mas não podem exportá-la ou excluí-la nem compartilhá-la com outros usuários.

Para conseguir compartilhar uma chave, faça login no HSM como o usuário de criptografia (CU) que tem a chave.

Tópicos

- [Exemplo: compartilhar e cancelar o compartilhamento de uma chave](#)
- [Tópicos relacionados](#)

Exemplo: compartilhar e cancelar o compartilhamento de uma chave

Example

O exemplo a seguir mostra como compartilhar e cancelar o compartilhamento de uma chave com o usuário de criptografia (CU) `alice`. Junto com os comandos `key share` e `key unshare`, os comandos de compartilhar e cancelar o compartilhamento também exigem uma chave específica usando [filtros de chave CLI do CloudHSM](#) e o nome de usuário específico do usuário com quem a chave será compartilhada ou não compartilhada.

1. Comece executando o `key list` comando com um filtro para retornar uma chave específica e ver com quem a chave já está compartilhada.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            }
          ]
        }
      }
    ]
  }
}
```

```
    },
    {
      "username": "cu1",
      "key-coverage": "full"
    },
    {
      "username": "cu4",
      "key-coverage": "full"
    },
    {
      "username": "cu5",
      "key-coverage": "full"
    },
    {
      "username": "cu6",
      "key-coverage": "full"
    },
    {
      "username": "cu7",
      "key-coverage": "full"
    },
  ],
  "key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
  },
  "cluster-coverage": "full"
},
"attributes": {
  "key-type": "rsa",
  "label": "rsa_key_to_share",
  "id": "",
  "check-value": "0xae8ff0",
  "class": "private-key",
  "encrypt": false,
  "decrypt": true,
  "token": true,
  "always-sensitive": true,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": true,
  "modifiable": true,
  "never-extractable": false,
```

```

        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": true,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1219,
        "public-exponent": "0x010001",
        "modulus":
"0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
        "modulus-size-bits": 2048
    }
  ],
  "total_key_count": 1,
  "returned_key_count": 1
}
}

```

2. Visualize a `shared-users` saída para identificar com quem a chave está compartilhada atualmente.
3. Para compartilhar essa chave com o usuário de criptografia (CU) `alice`, digite o seguinte comando:

```

aws-cloudhsm > key share --filter attr.label="rsa_key_to_share" attr.class=private-
key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "message": "Key shared successfully"
  }
}

```

Observe que, junto com o `key share` comando, esse comando usa o rótulo exclusivo da chave e o nome do usuário com quem a chave será compartilhada.

4. Execute o comando `key list` para confirmar que a chave foi compartilhada com `alice`:

```

aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" --verbose
{

```

```
"error_code": 0,
"data": {
  "matched_keys": [
    {
      "key-reference": "0x000000000001c0686",
      "key-info": {
        "key-owners": [
          {
            "username": "cu3",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu1",
            "key-coverage": "full"
          },
          {
            "username": "cu4",
            "key-coverage": "full"
          },
          {
            "username": "cu5",
            "key-coverage": "full"
          },
          {
            "username": "cu6",
            "key-coverage": "full"
          },
          {
            "username": "cu7",
            "key-coverage": "full"
          },
          {
            "username": "alice",
            "key-coverage": "full"
          }
        ]
      },
      "key-quorum-values": {
        "manage-key-quorum-value": 0,
```

```

        "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
},
"attributes": {
    "key-type": "rsa",
    "label": "rsa_key_to_share",
    "id": "",
    "check-value": "0xae8ff0",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1219,
    "public-exponent": "0x010001",
    "modulus":
"0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
    "modulus-size-bits": 2048
}
}
],
"total_key_count": 1,
"returned_key_count": 1
}
}

```

5. Para cancelar o compartilhamento da mesma chave com alice, execute o seguinte unshare comando:

```
aws-cloudhsm > key unshare --filter attr.label="rsa_key_to_share"
attr.class=private-key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "message": "Key unshared successfully"
  }
}
```

Observe que, junto com o `key unshare` comando, esse comando usa o rótulo exclusivo da chave e o nome do usuário com quem a chave será compartilhada.

6. Execute o `key list` comando novamente e confirme se a chave não foi compartilhada com o usuário de criptografia `alice`:

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            }
          ]
        }
      }
    ]
  }
}
```

```
{
  "username": "cu5",
  "key-coverage": "full"
},
{
  "username": "cu6",
  "key-coverage": "full"
},
{
  "username": "cu7",
  "key-coverage": "full"
},
],
"key-quorum-values": {
  "manage-key-quorum-value": 0,
  "use-key-quorum-value": 0
},
"cluster-coverage": "full"
},
"attributes": {
  "key-type": "rsa",
  "label": "rsa_key_to_share",
  "id": "",
  "check-value": "0xae8ff0",
  "class": "private-key",
  "encrypt": false,
  "decrypt": true,
  "token": true,
  "always-sensitive": true,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": true,
  "modifiable": true,
  "never-extractable": false,
  "private": true,
  "sensitive": true,
  "sign": true,
  "trusted": false,
  "unwrap": true,
  "verify": false,
  "wrap": false,
  "wrap-with-trusted": false,
  "key-length-bytes": 1219,
```

```
        "public-exponent": "0x010001",
        "modulus":
        "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254",
        "modulus-size-bits": 2048
    }
}
],
"total_key_count": 1,
"returned_key_count": 1
}
}
```

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Compartilhar uma chave usando a CloudHSM CLI](#)
- [Cancelar o compartilhamento de uma chave usando a CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

Filtrar chaves usando a CLI do CloudHSM

Use os seguintes comandos deste tópico para utilizar os mecanismos padronizados de filtragem de chaves para a CLI do CloudHSM.

- key list
- key delete
- key share
- key unshare
- key set-attribute

Para selecionar chaves de and/or filtro com a CLI do CloudHSM, os comandos principais utilizam um mecanismo de filtragem padronizado baseado em. [Atributos de chave da CloudHSM CLI](#) Uma chave ou conjunto de teclas pode ser especificado em comandos de teclas usando um ou mais AWS CloudHSM atributos que podem identificar uma única chave ou várias chaves. O mecanismo de filtragem de chaves opera somente nas chaves que o usuário atualmente conectado possui e compartilha, bem como em todas as chaves públicas no AWS CloudHSM cluster.

Tópicos

- [Requisitos](#)
- [Filtrando para encontrar uma única chave](#)
- [Erros de filtragem](#)
- [Tópicos relacionados](#)

Requisitos

Para filtrar as chaves, você deve estar logado como usuário criptográfico (CUs).

Filtrando para encontrar uma única chave

Observe que, nos exemplos a seguir, cada atributo usado como filtro deve ser escrito na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE`. Por exemplo, se você quiser filtrar pelo atributo de rótulo, você escreverá `attr.label=my_label`.

Example Use um único atributo para encontrar uma única chave

Este exemplo demonstra como filtrar para uma única chave exclusiva usando somente um único atributo de identificação.

```
aws-cloudhsm > key list --filter attr.label="my_unique_key_label" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "alice",
              "key-coverage": "full"
            }
          ]
        }
      }
    ]
  }
}
```

```

    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "rsa",
    "label": "my_unique_key_label",
    "id": "",
    "check-value": "0xae8ff0",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1219,
    "public-exponent": "0x010001",
    "modulus":
      "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254c8f5",
    "modulus-size-bits": 2048
  }
},
"total_key_count": 1,
"returned_key_count": 1
}
}

```

Example Use múltiplos atributos para encontrar uma única chave

O exemplo a seguir demonstra como encontrar uma única chave usando vários atributos de chave.

```
aws-cloudhsm > key list --filter attr.key-type=rsa attr.class=private-key attr.check-  
value=0x29bbd1 --verbose  
{  
  "error_code": 0,  
  "data": {  
    "matched_keys": [  
      {  
        "key-reference": "0x00000000000540011",  
        "key-info": {  
          "key-owners": [  
            {  
              "username": "cu3",  
              "key-coverage": "full"  
            }  
          ],  
          "shared-users": [  
            {  
              "username": "cu2",  
              "key-coverage": "full"  
            }  
          ],  
          "key-quorum-values": {  
            "manage-key-quorum-value": 0,  
            "use-key-quorum-value": 0  
          },  
          "cluster-coverage": "full"  
        },  
        "attributes": {  
          "key-type": "rsa",  
          "label": "my_crypto_user",  
          "id": "",  
          "check-value": "0x29bbd1",  
          "class": "my_test_key",  
          "encrypt": false,  
          "decrypt": true,  
          "token": true,  
          "always-sensitive": true,  
          "derive": false,  
          "destroyable": true,  
          "extractable": true,  
        }  
      }  
    ]  
  }  
}
```

```

        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": true,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
"0x8b3a7c20618e8be08220ed8ab2c8550b65fc1aad8d4cf04fbf2be685f97eeb78fcbbad9b02cd91a3b15e990c2a7
        "modulus-size-bits": 2048
    }
}
],
"total_key_count": 1,
"returned_key_count": 1
}
}

```

Example Como filtrar para encontrar um conjunto de chaves

O exemplo a seguir demonstra como filtrar para encontrar um conjunto de chaves rsa privadas.

```

aws-cloudhsm > key list --filter attr.key-type=rsa attr.class=private-key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "my_crypto_user",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {

```

```

        "username": "cu2",
        "key-coverage": "full"
    },
    {
        "username": "cu1",
        "key-coverage": "full"
    },
],
"key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
},
"cluster-coverage": "full"
},
"attributes": {
    "key-type": "rsa",
    "label": "rsa_key_to_share",
    "id": "",
    "check-value": "0xae8ff0",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1219,
    "public-exponent": "0x010001",
    "modulus":
"0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254c8f5",
    "modulus-size-bits": 2048
}

```

```
},
{
  "key-reference": "0x00000000000540011",
  "key-info": {
    "key-owners": [
      {
        "username": "my_crypto_user",
        "key-coverage": "full"
      }
    ],
    "shared-users": [
      {
        "username": "cu2",
        "key-coverage": "full"
      }
    ],
    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "rsa",
    "label": "my_test_key",
    "id": "",
    "check-value": "0x29bbd1",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
```

```

        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
"0x8b3a7c20618e8be08220ed8ab2c8550b65fc1aad8d4cf04fbf2be685f97eeb78fcbbad9b02cd91a3b15e990c2a7
        "modulus-size-bits": 2048
    }
}
],
"total_key_count": 2,
"returned_key_count": 2
}
}

```

Erros de filtragem

Certas operações de chaves só podem ser executadas em uma única chave por vez. Para essas operações, a CLI do CloudHSM fornecerá um erro se os critérios de filtragem não forem suficientemente refinados e várias chaves corresponderem aos critérios. Um exemplo é mostrado abaixo com a chave delete.

Exemplo Erro de filtragem ao combinar muitas chaves

```

aws-cloudhsm > key delete --filter attr.key-type=rsa
{
  "error_code": 1,
  "data": "Key selection criteria matched 48 keys. Refine selection criteria to select
a single key."
}

```

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)

Marcar uma chave como confiável usando a CLI do CloudHSM

O conteúdo desta seção fornece instruções sobre como usar a CLI do CloudHSM para marcar uma chave como confiável.

1. Usando o comando [loginCLI do CloudHSM](#), faça login como um usuário de criptografia (CU).

2. Use o `key list` comando para identificar a referência chave da chave que você deseja marcar como confiável. O exemplo a seguir lista a chave com o rótulo `key_to_be_trusted`.

```
aws-cloudhsm > key list --filter attr.label=test_aes_trusted
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000200333",
        "attributes": {
          "label": "test_aes_trusted"
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Usando o comando [Sair de um HSM usando a CLI do CloudHSM](#), faça logout como um usuário de criptografia (CU).
4. Usando o [Fazer login em um HSM usando a CLI do CloudHSM](#) comando, faça login como administrador.
5. Usando o comando [key set-attribute \(atributo de configuração de chave\)](#) com a referência de chave que você identificou na etapa 2, defina o valor confiável da chave como verdadeiro:

```
aws-cloudhsm > key set-attribute --filter key-reference=<Key Reference> --name
trusted --value true
{
  "error_code": 0,
  "data": {
    "message": "Attribute set successfully"
  }
}
```


Gerenciar a autenticação de quórum (controle de acesso M ou N) usando a CLI do CloudHSM

Os módulos de segurança de hardware (HSMs) em seu AWS CloudHSM cluster oferecem suporte à autenticação de quorum, também conhecida como controle de acesso M of N. Com a autenticação de quórum, nenhum usuário único no HSM pode realizar operações controladas pelo quórum. Em vez disso, um número mínimo de usuários do HSM (pelo menos 2) deve cooperar para realizar essas operações. A autenticação de quórum adiciona uma camada adicional de proteção, exigindo aprovações de vários usuários do HSM.

A autenticação de quorum pode controlar as seguintes operações:

- Uso e gerenciamento de chaves HSM por um [usuário de criptografia](#): criação de assinaturas com uma chave ou encapsulamento, desencapsulamento, compartilhamento, descompartilhamento e configuração de um atributo de uma chave.

Considerações importantes

- Um usuário do HSM pode assinar seu próprio token de quorum, ou seja, o usuário solicitante pode fornecer uma das aprovações necessárias para a autenticação de quorum.
- Você escolhe o número mínimo de aprovadores de quorum para operações controladas por quorum. O menor número que é possível escolher é dois (2) e o maior número que é possível escolher é oito (8).
- O HSM pode armazenar até 1.024 tokens de quórum. Se o HSM já tiver 1.024 tokens quando você tentar criar um novo, ele removerá um dos tokens expirados. Por padrão, tokens expiram dez minutos após sua criação.
- Se a autenticação multifator (MFA) estiver ativada, o cluster usará a mesma chave para autenticação de quórum e para MFA. Para obter mais informações sobre o uso da autenticação de quórum e da MFA, consulte [Usar a CloudHSM CLI para gerenciar a MFA](#).
- Cada HSM pode conter apenas um token por serviço administrativo por vez, mas vários tokens por serviço Usuário de criptografia.

Os seguintes tópicos fornecem mais informações sobre a autenticação de quorum no AWS CloudHSM.

Tópicos

- [Processo de autenticação de quórum para a CLI do CloudHSM](#)
- [Nomes e tipos AWS CloudHSM de serviços compatíveis para autenticação de quórum com o CloudHSM CLI](#)
- [Configure a autenticação de quórum para AWS CloudHSM usuários de criptografia usando o CloudHSM CLI](#)
- [Gerenciamento e uso de chaves com autenticação de quorum habilitada para usar a CLI do AWS CloudHSM CloudHSM](#)

Processo de autenticação de quórum para a CLI do CloudHSM

As etapas a seguir resumem os processos de autenticação de quórum para a CLI do CloudHSM. Para conhecer as etapas e ferramentas específicas, consulte [Gerenciamento e uso de chaves com autenticação de quorum habilitada para usar a CLI do AWS CloudHSM CloudHSM](#).

1. Cada usuário do módulo de segurança de hardware (HSM) cria uma chave assimétrica para assinatura. Isso é feito fora do HSM, tomando cuidado para proteger a chave apropriadamente.
2. Cada usuário do HSM faz login no HSM e registra a parte pública de sua chave de assinatura (a chave pública) no HSM.
3. Quando um usuário do HSM quer fazer uma operação controlada por quorum, ele faz login no HSM e obtém um token de quorum.
4. O usuário do HSM fornece esse token de quorum a um ou mais usuários do HSM e solicita sua aprovação.
5. Os outros usuários do HSM aprovam usando suas chaves para assinar criptograficamente o token de quorum. Isso ocorre fora do HSM.
6. Quando o usuário do HSM tem o número necessário de aprovações, ele faz login no HSM e executa a operação controlada por quórum com o argumento `--approval`, fornecendo o arquivo de token de quórum assinado, que contém todas as aprovações (assinaturas) necessárias.
7. O HSM usa as chaves públicas registradas de cada assinante para verificar as assinaturas. Se as assinaturas forem válidas, o HSM aprova o token e a operação controlada por quórum será executada.

Nomes e tipos AWS CloudHSM de serviços compatíveis para autenticação de quórum com o CloudHSM CLI

Serviços administrativos: a autenticação de quórum é usada para serviços com privilégios administrativos, como criar usuários, excluir usuários, alterar senhas de usuários, definir valores de quórum e desativar recursos de quórum e MFA.

Serviços de criptografia para usuários: a autenticação de quórum é usada para serviços privilegiados de usuários criptográficos associados a uma chave específica, como assinar com uma chave, uma chave, sharing/unsharing uma chave e definir o atributo de wrapping/unwrapping uma chave. O valor do quórum de uma chave associada é configurado quando a chave é gerada, importada ou desencapsulada. O valor do quórum deve ser igual ou menor do que o número de usuários aos quais a chave está associada, o que inclui usuários com os quais a chave é compartilhada e o proprietário da chave.

Cada tipo de serviço é subdividido em um nome de serviço qualificado, que contém um conjunto específico de operações de serviço suportadas por quórum que podem ser executadas.

Nome do serviço	Tipo de serviço	Operações de serviço
usuário	Administrador	- criar usuário - excluir usuário - alterar senha de usuário - user change-mfa
quorum	Administrador	sinal simbólico de quórum set-quorum-value
cluster ¹	Administrador	- cluster mtls register-trust-anchor - cluster mtls deregister-trust-anchor - cluster mtls set-enforcement
gerenciamento de chaves	Usuário de criptografia	- encapsulamento de chaves - desencapsulamento de chaves

Nome do serviço	Tipo de serviço	Operações de serviço
		• compartilhamento de chaves • descompartilhamento de chaves • key set-attribute
uso de chave	Usuário de criptografia	• sinal de chave

[1] O serviço de cluster está disponível exclusivamente em instâncias hsm2m.medium

Configure a autenticação de quórum para AWS CloudHSM usuários de criptografia usando o CloudHSM CLI

Estes tópicos descrevem como configurar seu CloudHSM para autenticação de quórum por [usuários de criptografia](#). Execute estas etapas uma vez durante a configuração inicial. Para gerenciamento e uso subsequentes de chaves, consulte [Gerenciamento e uso de chaves com autenticação de quorum habilitada para usar a CLI do AWS CloudHSM CloudHSM](#).

Tópicos

- [Pré-requisitos](#)
- [Etapa 1. Criar e registrar uma chave para assinatura](#)
- [Etapa 2. Definir os principais valores do quórum durante a geração da chave](#)

Pré-requisitos

- Familiaridade com a [CloudHSM CLI](#)

Etapa 1. Criar e registrar uma chave para assinatura

Para usar a autenticação de quórum, cada usuário de criptografia deve concluir todas as etapas a seguir:

Tópicos

- [Criar um par de chaves RSA](#)

- [Criar um token de registro](#)
- [Assinar o token de registro não assinado](#)
- [Registrar uma chave pública no HSM](#)

Criar um par de chaves RSA

Existem muitas maneiras diferentes de criar e proteger um par de chaves. O exemplo a seguir mostra como fazer isso com o [OpenSSL](#).

Exemplo Crie uma chave privada com o OpenSSL

O exemplo a seguir demonstra como usar o OpenSSL para criar uma chave RSA de 2048 bits. Para usar esse exemplo, *<crypto_user1.key>* substitua pelo nome do arquivo em que você deseja armazenar a chave.

```
$ openssl genrsa -out <crypto_user1.key>
Generating RSA private key, 2048 bit long modulus
.....+++
.+++
e is 65537 (0x10001)
```

Em seguida, gere a chave pública usando a chave privada que você acabou de criar.

Exemplo Crie uma chave pública com o OpenSSL

O exemplo a seguir demonstra como usar o OpenSSL para criar uma chave pública a partir da chave privada que você acabou de criar.

```
$ openssl rsa -in crypto_user1.key -outform PEM -pubout -out crypto_user1.pub
writing RSA key
```

Criar um token de registro

Você cria um token e o assina com a chave privada que acabou de gerar na etapa anterior.

Criar um token de registro

1. Use o comando a seguir para iniciar CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Crie um token de registro executando o comando [quorum token-sign generate](#):

```
aws-cloudhsm > quorum token-sign generate --service registration --token /path/
tokenfile
{
  "error_code": 0,
  "data": {
    "path": "/path/tokenfile"
  }
}
```

3. O comando [quorum token-sign generate](#) gera um token de registro no caminho de arquivo especificado. Inspeção o arquivo de token:

```
$ cat /path/tokenfile
{
  "version": "2.0",
  "tokens": [
    {
      "approval_data": <approval data in base64 encoding>,
      "unsigned": <unsigned token in base64 encoding>,
      "signed": ""
    }
  ]
}
```

O caminho do arquivo consiste no seguinte:

- approval_data: um token de dados aleatório codificado em base64 cujos dados brutos não excedem o máximo de 245 bytes.
- unsigned: um token codificado em base64 e com SHA256 hash do approval_data.

- **signed**: um token assinado codificado em base64 (assinatura) do token não assinado, usando a chave privada RSA de 2048 bits gerada anteriormente com o OpenSSL.

Você assina o token não assinado com a chave privada para demonstrar que tem acesso à chave privada. Você precisará do arquivo do token de registro totalmente preenchido com uma assinatura e a chave pública para registrar o usuário criptográfico como usuário de quórum no cluster. AWS CloudHSM

Assinar o token de registro não assinado

1. Decodifique o token não assinado codificado em base64 e coloque-o em um arquivo binário:

```
$ echo -n '6BMUj6mUjjko6ZLCEdzG1WpR5sILhFJfqhW1ej30q1g=' | base64 -d >
crypto_user.bin
```

2. Use OpenSSL e a chave privada para assinar o token de registro não assinado, agora binário, e criar um arquivo de assinatura binária:

```
$ openssl pkeyutl -sign \
-inkey crypto_user1.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in crypto_user.bin \
-out crypto_user.sig.bin
```

3. Codifique a assinatura binária em base64:

```
$ base64 -w0 crypto_user.sig.bin > crypto_user.sig.b64
```

4. Copie e cole a assinatura codificada em base64 no arquivo de token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "approval_data": <approval data in base64 encoding>,
      "unsigned": <unsigned token in base64 encoding>,
      "signed": <signed token in base64 encoding>
    }
  ]
}
```

```
}
```

Registrar uma chave pública no HSM

Depois de criar uma chave, o usuário criptográfico deve registrar a chave pública no AWS CloudHSM cluster.

1. Inicie a CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Faça login como o usuário de criptografia cuja chave pública você deseja registrar.

```
aws-cloudhsm > login --username crypto_user1 --role crypto-user
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "crypto_user1",
    "role": "crypto-user"
  }
}
```

3. Registre uma chave pública no [Registrar a estratégia de quórum de assinatura de token de um usuário usando a CLI do CloudHSM](#). Para obter mais informações, consulte o exemplo a seguir ou use o comando `help user change-quorum token-sign register`.

Example— Registrar uma chave pública com o AWS CloudHSM cluster

O exemplo a seguir mostra como usar o comando `user change-quorum token-sign register` na CloudHSM CLI para registrar a chave pública de um administrador no HSM. Para usar esse comando, o usuário de criptografia deve estar conectado no HSM. Substitua estes valores pelos seus próprios:


```
aws-cloudhsm > user change-quorum token-sign register --public-key </path/
crypto_user.pub> --signed-token </path/tokenfile>
{
  "error_code": 0,
  "data": {
    "username": "crypto_user1",
    "role": "crypto-user"
  }
}
```

Note

/path/crypto_user.pub: o caminho do arquivo PEM de chave pública

Obrigatório: sim

/path/token-file: o caminho do arquivo com token assinado pela chave privada do usuário

Obrigatório: sim

4. Depois que todos os usuários de criptografia registrarem suas chaves públicas, a saída do comando `user list` mostra essa informação no campo de quórum, informando a estratégia de quórum ativada em uso.

Neste exemplo, o AWS CloudHSM cluster tem dois HSMS, cada um com os mesmos usuários de criptografia, conforme mostrado na saída a seguir do `user list` comando. Para obter mais informações sobre como criar usuários, consulte [Gerenciamento de usuários com a CLI do CloudHSM](#).

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
```

```
"username": "crypto_user1",
"role": "crypto-user",
"locked": "false",
"mfa": [],
"quorum": [
  {
    "strategy": "token-sign",
    "status": "enabled"
  }
],
"cluster-coverage": "full"
},
{
  "username": "crypto_user2",
  "role": "crypto-user",
  "locked": "false",
  "mfa": [],
  "quorum": [
    {
      "strategy": "token-sign",
      "status": "enabled"
    }
  ],
  "cluster-coverage": "full"
},
{
  "username": "crypto_user3",
  "role": "crypto-user",
  "locked": "false",
  "mfa": [],
  "quorum": [
    {
      "strategy": "token-sign",
      "status": "enabled"
    }
  ],
  "cluster-coverage": "full"
},
{
  "username": "app_user",
  "role": "internal(APPLIANCE_USER)",
  "locked": "false",
  "mfa": [],
  "quorum": [],
```

```
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Etapa 2. Definir os principais valores do quórum durante a geração da chave

Para usar a autenticação de quórum, é preciso que um usuário de criptografia faça login no HSM e, em seguida, defina os valores de quórum chave associados. Esse é o número mínimo de aprovações de usuários de criptomoedas necessárias para realizar as principais operações do HSM. management/usage Para saber mais sobre os comandos de teclas associados ao gerenciamento ou ao uso de chaves, consulte [Serviços e tipos compatíveis](#).

Gerar um par de chaves com valores de quórum chave definidos

1. Use o comando a seguir para iniciar a CloudHSM CLI:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Usando a CloudHSM CLI, faça login como um usuário de criptografia.

```
aws-cloudhsm > login --username crypto_user1 --role crypto-user
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "crypto_user1",
    "role": "crypto-user"
  }
}
```

Este exemplo gera um par de chaves RSA com os valores de quórum de chaves dois (2) definidos para operações de gerenciamento e uso de chaves. Você pode escolher qualquer valor de zero (0) até oito (8), até o número total de usuários de criptografia no HSM. Neste exemplo, o HSM tem três (3) usuários de criptografia, então o valor máximo possível é três (3). Observe que, neste exemplo, estamos compartilhando a chave *<crypto_user2>* durante a geração da chave. Observe também que as chaves públicas não têm valores de quórum.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes verify=true \
--private-attributes sign=true
--share-crypto-users crypto_user2 \
--manage-private-key-quorum-value 2 \
--use-private-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000000640006",
      "key-info": {
        "key-owners": [
          {
            "username": "crypto_user",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "0x",
        "check-value": "0x218f50",
        "class": "public-key",
        "encrypt": false,
```

```

    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
"0xbdf471a3d2a869492f51c767bece8780730ae6479a9a75efffe7cea3594fb28ca518630e7b1d988b45d2fedc830
    "modulus-size-bits": 2048
  }
},
"private_key": {
  "key-reference": "0x00000000000640007",
  "key-info": {
    "key-owners": [
      {
        "username": "crypto_user",
        "key-coverage": "full"
      }
    ],
    "shared-users": [
      {
        "username": "crypto_user2",
        "key-coverage": "full"
      }
    ],
    "key-quorum-values": {
      "manage-key-quorum-value": 2,
      "use-key-quorum-value": 2
    },
    "cluster-coverage": "full"
  }
}

```

```

    },
    "attributes": {
      "key-type": "rsa",
      "label": "rsa-private-key-example",
      "id": "0x",
      "check-value": "0x218f50",
      "class": "private-key",
      "encrypt": false,
      "decrypt": false,
      "token": true,
      "always-sensitive": true,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": true,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": true,
      "trusted": false,
      "unwrap": false,
      "verify": false,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 1216,
      "public-exponent": "0x010001",
      "modulus":
"0xbdf471a3d2a869492f51c767bece8780730ae6479a9a75efffe7cea3594fb28ca518630e7b1d988b45d2fedc830
      "modulus-size-bits": 2048
    }
  }
}
}

```

Ao gerar uma chave com controles de quórum, a chave deve estar associada a um número mínimo de usuários igual ao maior valor de quórum da chave. Os usuários associados incluem o proprietário da chave e os usuários de criptografia com quem essa chave é compartilhada. Para definir o número mínimo de usuários com quem compartilhar a chave, obtenha o maior valor de quórum entre o valor do quórum de uso da chave e o valor do quórum de gerenciamento de chaves e subtraia 1 para contabilizar o proprietário da chave, que, por padrão, está associado à chave. Para compartilhar a chave com mais usuários, use o comando [Compartilhar uma chave usando a CloudHSM CLI](#).

A falha em compartilhar a chave com usuários suficientes na geração da chave resultará em falha, conforme mostrado abaixo.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes verify=true \
--private-attributes sign=true \
--share-crypto-users crypto_user2 crypto_user3 \
--manage-private-key-quorum-value 3 \
--use-private-key-quorum-value 4
{
  "error_code": 1,
  "data": "Invalid quorum value provided."
}
```

Gerenciamento e uso de chaves com autenticação de quorum habilitada para usar a CLI do AWS CloudHSM CloudHSM

Depois de configurar a autenticação de quórum para seu AWS CloudHSM cluster, os usuários de criptomoedas não poderão realizar operações de gerenciamento ou uso de chaves do HSM sozinhos se a chave tiver valores de quórum associados. Este tópico explica como um usuário de criptografia pode obter um token temporário para realizar uma operação de gerenciamento ou uso de chaves do HSM.

Note

Cada token de quórum é válido para uma operação. Quando essa operação é bem-sucedida, o token não é mais válido e o usuário de criptografia deve obter um novo token. Um token de quórum só é válido durante sua sessão de login atual. Se você fizer log out da CloudHSM CLI ou se a rede se desconectar, o token não será mais válido e você precisará obter um novo. Você pode usar somente um token do CloudHSM na CloudHSM CLI. Ele não pode ser usado para autenticação em uma aplicação diferente.

O exemplo a seguir mostra a saída quando um usuário de criptografia tenta criar uma assinatura com uma chave associada ao quórum no HSM após a configuração da autenticação de quórum. O comando falha com um erro `Quorum Failed`, o que significa que a autenticação de quórum falhou:

```
aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private-key-example --  
hash-function sha256 --data YWJjMTIz  
{  
  "error_code": 1,  
  "data": "Quorum Failed"  
}
```

Um usuário de criptografia deve concluir as seguintes tarefas para obter um token temporário a fim de realizar uma operação de gerenciamento ou uso de chaves no HSM:

Etapas

- [Etapa 1. Obter um token de quorum](#)
- [Etapa 2. Obter assinaturas dos usuários de criptografia de aprovação](#)
- [Etapa 3. Aprove o token no cluster do CloudHSM e execute uma operação](#)

Etapa 1. Obter um token de quorum

1. Inicie a CloudHSM CLI.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Faça login no cluster como usuário de criptografia.

```
aws-cloudhsm > login --username <crypto_user1> --role crypto-user --  
password password123
```

Este exemplo faz login de `crypto_user1` na CloudHSM CLI com a função `crypto-user`. Substitua estes valores pelos seus próprios.

```
{  
  "error_code": 0,  
  "data": {  
    "username": "crypto_user1",
```



```

    "role": "crypto-user"
  }
}

```

3. Use o comando `quorum token-sign generate` para obter um token de quórum.

No comando a seguir, `key-usage` identifica o nome do serviço em que você pode usar o token que você está gerando. Nesse caso, o token é para operações de uso de chaves (serviço `key-usage`). Este exemplo usa o sinalizador `--filter` para associar o token a uma chave específica.

```

aws-cloudhsm > quorum token-sign generate --service key-usage --token </path/
crypto_user1.token> --filter attr.label=rsa-private-key-example
{
  "error_code": 0,
  "data": {
    "path": "/home/crypto_user1.token"
  }
}

```

Este exemplo obtém um token de quórum para o usuário de criptografia com o nome de usuário `crypto_user1` e salva o token em um arquivo chamado `crypto_user1.token`. Para usar o exemplo de comando, substitua esses valores por seus próprios:

O comando `quorum token-sign generate` gera um token de quórum de serviço de uso de chave no caminho de arquivo especificado. Você pode inspecionar o arquivo de token:

```

$ cat </path/crypto_user1.token>
{
  "version": "2.0",
  "service": "key-usage",
  "key_reference": "0x00000000000680006",
  "approval_data":
  "AAIABQAAABkAAAAAGgABi5CDa9x9VyyRIaFbkSrHgJjcnlwdG9fdXNlcmgAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
+GJj8gXo9lKuANGNyeXB0b191c2VyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGc9AEsAAAAAAAAAAAAAA==",
  "token": "5G1goW0lQU4fw4QI1bXkPGZV0VoDugFGuSKE/k67ncM=",
  "signatures": []
}

```

O caminho do arquivo consiste no seguinte:

- **service**: um identificador do serviço de quórum ao qual o token está associado.
- **key_reference**: um identificador para a chave à qual esse token de quórum está associado.
- **approval_data**: um token de dados brutos codificado em base64 gerado pelo HSM.
- **token**: um token codificado em base64 e com hash SHA-256 do **approval_data**
- **signatures**: uma matriz de tokens assinados codificados em base64 (assinaturas) do token não assinado. Cada assinatura do aprovador está na forma de um literal de objeto JSON:

```
{
  "username": "<APPROVER_USERNAME>",
  "role": "<APPROVER_ROLE>",
  "signature": "<APPROVER_RSA2048_BIT_SIGNATURE>"
}
```

Cada assinatura é criada do resultado de um aprovador usando sua chave privada RSA de 2048 bits correspondente, cuja chave pública foi registrada no HSM.

4. Valide o novo token de quórum do serviço de usuário. O comando `quorum token-sign list` confirma que o token existe no CloudHSM.

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": [
      {
        "username": "crypto_user",
        "service": "key-usage",
        "key-reference": "0x00000000000680006",
        "minimum-token-count": 2
      }
    ]
  }
}
```

O **minimum-token-count** apresenta uma visão agregada do cluster do número mínimo utilizável de tokens de chave correspondentes ao nome de usuário, serviço e referência de chave recuperados de um único HSM no cluster.

Por exemplo, considerando um cluster de 2 HSM, se recebermos dois (2) tokens de uso de chave gerados pelo usuário `crypto_user1` para chave com referência `0x0000000000680006` do primeiro HSM no cluster e recebermos um (1) token de uso de chave gerado pelo usuário `crypto_user1` para chave com referência `0x0000000000680006` do outro HSM no cluster, exibiremos `"minimum-token-count": 1`.

Etapa 2. Obter assinaturas dos usuários de criptografia de aprovação

Um usuário de criptografia que tenha um token de quórum deve fazer com que o token seja aprovado por outros usuários de criptografia. Para darem sua aprovação, os outros usuários de criptografia usam suas chaves de assinatura para assinar criptograficamente o token fora do HSM.

Existem muitas maneiras diferentes de assinar o token. O exemplo a seguir mostra como assinar o token usando [OpenSSL](#). Para usar uma ferramenta de assinatura diferente, certifique-se de que essa ferramenta use a chave privada (chave de assinatura) do usuário de criptografia para assinar um resumo SHA-256 do token.

Neste exemplo, o usuário de criptografia que possui o token (`crypto-user`) precisa de pelo menos duas (2) aprovações. Os comandos do exemplo a seguir mostram como dois (2) usuários de criptografia podem usar o OpenSSL para assinar criptograficamente o token.

1. Decodifique o token não assinado codificado em base64 e coloque-o em um arquivo binário:

```
$echo -n '5G1goW01QU4fw4QI1bXkPGZV0VoDugFGuSKE/k67ncM=' | base64 -d >
crypto_user1.bin
```

2. Use o OpenSSL e a chave privada do aprovador para assinar o token não assinado de quórum binário para o serviço ao usuário e criar um arquivo de assinatura binária:

```
$openssl pkeyutl -sign \
-inkey crypto_user1.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in crypto_user1.bin \
-out crypto_user1.sig.bin
```

3. Codifique a assinatura binária em base64:

```
$ base64 -w0 crypto_user1.sig.bin > crypto_user1.sig.b64
```

4. Copie e cole a assinatura codificada em base64 no arquivo de token, usando o formato literal do objeto JSON especificado anteriormente para a assinatura do aprovador:

```
{
  "version": "2.0",
  "service": "key-usage",
  "key_reference": "0x00000000000680006",
  "approval_data":
  "AAIABQAAABkAAAAAGgABi5CDa9x9VyyRIaFbkSrHgJjcnlwdG9fdXNlcmgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
+GJj8gXo9lKuANGNyeXB0b191c2VyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGc9AEsAAAAAAAAAAAAA==",
  "token": "5GlgoW0lQU4fw4QIlbXkPGZV0VoDugFGuSKE/k67ncM=",
  "signatures": [
    {
      "username": "crypto_user1",
      "role": "crypto-user",
      "signature": "wa7aPzmGwBjcEoZ6jAzYASp841Afg0vcI27Y/
tGlCj1E9DawnFw5Uf0IJT2Ca7T5XD2ThVkuI0B+dhAomdqYNl6aUUFrJyH9GBJ
+E0PmA5jNVm25tzeRWBjzneTg4/
zTeE2reNqrHFHicWnttQLe9jS09J1znuDGWDe0HaBKWUaz2gUInJRqmeXDsZYdSvZksrqUH5dci/
RsaDE2+tGiS9g0RcIkFbsPW4HpGe2e5HVzGsqrV803PKlYQv6+fymfcNTTuoxKcHAK0jpl43QSuSIu2gVq7KI8mSmmW
+oiukaNfLJr+MoDKzAvCGDg4cDArg=="
    },
    {
      "username": "crypto_user2",
      "role": "crypto-user",
      "signature": "wa7aPzmGwBjcEoZ6jAzYASp841Afg0vcI27Y/
tGlCj1E9DawnFw5Uf0IJT2Ca7T5XD2ThVkuI0B+dhAomdqYNl6aUUFrJyH9GBJ
+E0PmA5jNVm25tzeRWBjzneTg4/
zTeE2reNqrHFHicWnttQLe9jS09J1znuDGWDe0HaBKWUaz2gUInJRqmeXDsZYdSvZksrqUH5dci/
RsaDE2+tGiS9g0RcIkFbsPW4HpGe2e5HVzGsqrV803PKlYQv6+fymfcNTTuoxKcHAK0jpl43QSuSIu2gVq7KI8mSmmW
+oiukaNfLJr+MoDKzAvCGDg4cDArg=="
    }
  ]
}
```

Etapa 3. Aprove o token no cluster do CloudHSM e execute uma operação

Depois que um usuário de criptografia tiver as aprovações e assinaturas necessárias, ele poderá enviar esse token ao cluster do CloudHSM junto com uma operação de gerenciamento ou uso de chaves.

Certifique-se de que a operação da chave corresponda ao serviço de quórum apropriado associado ao token de quórum. Para obter mais informações, consulte [Serviços e tipos compatíveis](#).

Durante a transação, o token será aprovado no AWS CloudHSM cluster e executará a operação de chave solicitada. O sucesso da operação de chave depende tanto de um token de quórum válido aprovado quanto de uma operação válida de chave.

Example Gerar uma assinatura com o mecanismo RSA-PKCS

No comando do exemplo a seguir, um usuário de criptografia logado cria uma assinatura com uma chave no HSM:

```
aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private-key-example --  
hash-function sha256 --data YWJjMTIz --approval /path/crypto_user1.token  
  
{  
  "error_code": 0,  
  "data": {  
    "key-reference": "0x00000000000640007",  
    "signature":  
    "h6hMqXacBrT3x3MXV13RXHdQno0+IQ6iy0kVrGzo23+eoWT0ZZgrSpBCu5KcuP6IYYHw9goQ5CfPf4jI1n05m/  
IUJtF1A1lmcz0HjEy1CJ7ICXNReDRye0U8m43dkJzt00UdkbtkDJGAcxkbKHLZ02uWsGXaQ8b0KhoGwsRAHHF6nldTXquIC  
+pZmUS38ythybney94Wj6fzY0ER8v7VIY5ijQGa3LfxrjSG4aw6QijEEbno5LSf18ahEaVKmVEnDBL54tylCJBgvGsYSY9H  
TDd2wfvP4PaxbFRyyHaw=="  
  }  
}
```

Se o usuário de criptografia tentar realizar outra operação de uso da chave HSM com o mesmo token, ela falhará:

```
aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private-key-example --  
hash-function sha256 --data YWJjMTIz --approval /home/crypto_user1.token  
  
{  
  "error_code": 1,  
  "data": "Quorum approval is required for this operation"  
}
```

Para realizar outra operação de chave do HSM, o usuário de criptografia deve gerar um novo token de quórum, obter novas assinaturas dos aprovadores e executar a operação de chave desejada com o argumento `--approval` para fornecer o token de quórum.

Use quorum token-sign list para verificar o token disponível. Este exemplo mostra que o usuário de criptografia não tem tokens aprovados.

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": []
  }
}
```

Gerenciamento de chaves com o AWS CloudHSM KMU

Se estiver usando a [série de versões mais recente do SDK](#), use a [CLI do CloudHSM](#) para gerenciar as chaves em seu cluster. AWS CloudHSM

Se estiver usando a [série de versões anteriores do SDK](#), você pode gerenciar chaves nos módulos de segurança de hardware (HSM) em seu AWS CloudHSM cluster usando a ferramenta de linha de comando key_mgmt_util (KMU). Antes de gerenciar as chaves, você deve iniciar o AWS CloudHSM cliente, iniciar key_mgmt_util e fazer login no. HSMs Para obter mais informações, consulte [Conceitos básicos do key_mgmt_util](#).

- [O uso de chaves confiáveis](#) descreve como usar os atributos da biblioteca PKCS #11 e a CMU para criar chaves confiáveis para proteger os dados.
- A [geração de chaves](#) tem instruções sobre a geração de chaves, incluindo chaves simétricas, chaves RSA e chaves EC.
- [A importação de chaves](#) fornece detalhes sobre como os proprietários das chaves importam as chaves.
- [A exportação de chaves](#) fornece detalhes sobre como os proprietários das chaves exportam as chaves.
- [A exclusão de chaves](#) fornece detalhes de como os proprietários das chaves excluem as chaves.
- O [compartilhamento e o não compartilhamento de chaves](#) detalham como os proprietários compartilham e cancelam o compartilhamento das chaves.

Gere chaves com o AWS CloudHSM KMU

Para gerar chaves no módulo de segurança de hardware (HSM), use o comando em AWS CloudHSM `key_mgmt_util` (KMU) que corresponde ao tipo de chave que você deseja gerar.

Tópicos

- [Gere chaves simétricas com o AWS CloudHSM KMU](#)
- [Gere pares de chaves RSA com o AWS CloudHSM KMU](#)
- [Gere pares de chaves ECC \(criptografia de curva elíptica\) usando o KMU AWS CloudHSM](#)

Gere chaves simétricas com o AWS CloudHSM KMU

Use o [genSymKey](#) comando em AWS CloudHSM `key_mgmt_util` (KMU) para gerar AES e outros tipos de chaves simétricas para. AWS CloudHSM Para consultar todas as opções disponíveis, use o comando `genSymKey -h`.

O exemplo a seguir cria uma chave AES de 256 bits.

```
Command: genSymKey -t 31 -s 32 -l aes256
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Created.  Key Handle: 524295

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Gere pares de chaves RSA com o AWS CloudHSM KMU

Para gerar um par de chaves RSA para AWS CloudHSM, use o comando [genRSAKey Pair](#) em AWS CloudHSM `key_mgmt_util`. Para consultar todas as opções disponíveis, use o comando `genRSAKeyPair -h`.

O exemplo a seguir gera um par de chaves RSA de 2048 bits.

```
Command: genRSAKeyPair -m 2048 -e 65537 -l rsa2048
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3GenerateKeyPair:    public key handle: 524294    private key handle: 524296
```

Cluster Error Status

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Gere pares de chaves ECC (criptografia de curva elíptica) usando o KMU AWS CloudHSM

Para gerar um par de chaves ECC para AWS CloudHSM, use o comando [gen ECCKey Pair](#) em AWS CloudHSM key_mgmt_util. Para consultar todas as opções disponíveis, incluindo uma lista das curvas elípticas suportadas, use o comando `genECCKeyPair -h`.

O exemplo a seguir gera um par de chaves ECC usando a curva elíptica P-384 definida na [publicação FIPS 186-4 do NIST](#).

```
Command: genECCKeyPair -i 14 -l ecc-p384
```

```
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3GenerateKeyPair:    public key handle: 524297    private key handle: 524298
```

Cluster Error Status

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Importar chaves com o AWS CloudHSM KMU

Para importar chaves secretas, ou seja, chaves simétricas e chaves privadas assimétricas, para o módulo de segurança de hardware (HSM) usando o AWS CloudHSM key_mgmt_util, você deve primeiro criar uma chave de encapsulamento no HSM. Importe chaves públicas diretamente sem uma chave de encapsulamento.

Tópicos

- [Importe chaves secretas com a AWS CloudHSM KMU](#)
- [Importe chaves públicas com a AWS CloudHSM KMU](#)

Importe chaves secretas com a AWS CloudHSM KMU

Conclua as etapas a seguir para importar uma chave secreta para AWS CloudHSM usar o `key_mgmt_util` (KMU). Antes de importar uma chave secreta, salve-a em um arquivo. Salve chaves simétricas como bytes brutos e chaves assimétricas privadas no formato PEM.

Este exemplo mostra como importar uma chave secreta com texto sem formatação de um arquivo para o HSM. Para importar uma chave criptografada de um arquivo para o HSM, use o [unWrapKey](#) comando.

Para importar uma chave secreta

1. Use o [genSymKey](#) comando para criar uma chave de empacotamento. O comando a seguir cria uma chave de encapsulamento AES de 128 bits válida apenas para a sessão atual. Você pode usar uma chave de sessão ou uma chave persistente como chave de encapsulamento.

```
Command: genSymKey -t 31 -s 16 -sess -l import-wrapping-key  
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS  
  
Symmetric Key Created. Key Handle: 524299  
  
Cluster Error Status  
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

2. Use um dos seguintes comandos, de acordo com o tipo de chave secreta que estiver importando.
 - Para importar uma chave simétrica, use o comando [imSymKey](#). O comando a seguir importa uma chave AES a partir de um arquivo chamado `aes256.key` usando a chave de encapsulamento criada na etapa anterior. Para consultar todas as opções disponíveis, use o comando `imSymKey -h`.

```
Command: imSymKey -f aes256.key -t 31 -l aes256-imported -w 524299  
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS  
  
Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS  
  
Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS  
  
Symmetric Key Unwrapped. Key Handle: 524300
```

Cluster Error Status

Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

Node id 1 and err state 0x00000000 : HSM Return: SUCCESS

Node id 2 and err state 0x00000000 : HSM Return: SUCCESS

- Para importar uma chave privada assimétrica, use o comando [importPrivateKey](#). O comando a seguir importa uma chave privada a partir de um arquivo chamado `rsa2048.key` usando a chave de encapsulamento criada na etapa anterior. Para consultar todas as opções disponíveis, use o comando `importPrivateKey -h`.

Command: **importPrivateKey -f rsa2048.key -l rsa2048-imported -w 524299**

BER encoded key length is 1216

Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Private Key Unwrapped. Key Handle: 524301

Cluster Error Status

Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

Node id 1 and err state 0x00000000 : HSM Return: SUCCESS

Node id 2 and err state 0x00000000 : HSM Return: SUCCESS

Importe chaves públicas com a AWS CloudHSM KMU

Use o [importPubKey](#) comando no AWS CloudHSM key_mgmt_util (KMU) para importar uma chave pública. Para consultar todas as opções disponíveis, use o comando `importPubKey -h`.

O exemplo a seguir importa uma chave pública RSA a partir de um arquivo chamado `rsa2048.pub`.

Command: **importPubKey -f rsa2048.pub -l rsa2048-public-imported**

Cfm3CreatePublicKey returned: 0x00 : HSM Return: SUCCESS

Public Key Handle: 524302

Cluster Error Status

Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

Node id 1 and err state 0x00000000 : HSM Return: SUCCESS

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Exportar chaves com o AWS CloudHSM KMU

Para exportar chaves AWS CloudHSM secretas, ou seja, chaves simétricas e chaves privadas assimétricas, do módulo de segurança de hardware (HSM) usando o AWS CloudHSM `key_mgmt_util` (KMU), você deve primeiro criar uma chave de encapsulamento. Exporte chaves públicas diretamente sem uma chave de encapsulamento.

Somente o proprietário da chave pode exportá-la. Os usuários com quem essa chave é compartilhada podem usá-la em operações criptográficas, mas não podem exportá-la. Ao executar este exemplo, certifique-se de exportar uma chave que você criou.

Important

O [exSymKey](#) comando grava uma cópia em texto simples (não criptografado) da chave secreta em um arquivo. O processo de exportação requer uma chave de encapsulamento, mas a chave no arquivo não é uma chave encapsulada. Para exportar uma cópia encapsulada (criptografada) de uma chave, use o comando [wrapKey](#).

Tópicos

- [Exporte chaves secretas com a AWS CloudHSM KMU](#)
- [Exportar chaves públicas com a AWS CloudHSM KMU](#)

Exporte chaves secretas com a AWS CloudHSM KMU

Conclua as etapas a seguir para exportar uma chave secreta AWS CloudHSM usando o `key_mgmt_util` (KMU).

Para exportar uma chave secreta

1. Use o [genSymKey](#) comando para criar uma chave de empacotamento. O comando a seguir cria uma chave de encapsulamento AES de 128 bits válida apenas para a sessão atual.

```
Command: genSymKey -t 31 -s 16 -sess -l export-wrapping-key  
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Created. Key Handle: 524304
```

```
Cluster Error Status
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

2. Use um dos seguintes comandos, de acordo com o tipo de chave secreta que estiver exportando.

- Para exportar uma chave simétrica, use o [exSymKey](#) comando. O comando a seguir exporta uma chave AES para um arquivo chamado `aes256.key.exp`. Para consultar todas as opções disponíveis, use o comando `exSymKey -h`.

```
Command: exSymKey -k 524295 -out aes256.key.exp -w 524304
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
Wrapped Symmetric Key written to file "aes256.key.exp"
```

 Note

A saída do comando diz que uma "Chave simétrica encapsulada" está gravada no arquivo de saída. No entanto, o arquivo de saída contém uma chave com texto sem formatação (não encapsulada). Para exportar uma chave encapsulada (criptografada) para um arquivo, use o comando [wrapKey](#).

- Para exportar uma chave privada, use o comando `exportPrivateKey`. O comando a seguir exporta uma chave privada para um arquivo chamado `rsa2048.key.exp`. Para consultar todas as opções disponíveis, use o comando `exportPrivateKey -h`.

```
Command: exportPrivateKey -k 524296 -out rsa2048.key.exp -w 524304
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
PEM formatted private key is written to rsa2048.key.exp
```

Exportar chaves públicas com a AWS CloudHSM KMU

Use o `exportPubKey` comando no AWS CloudHSM `key_mgmt_util` (KMU) para exportar uma chave pública. Para consultar todas as opções disponíveis, use o comando `exportPubKey -h`.

O exemplo a seguir exporta uma chave pública RSA para um arquivo chamado `rsa2048.pub.exp`.

```
Command: exportPubKey -k 524294 -out rsa2048.pub.exp  
PEM formatted public key is written to rsa2048.pub.key  
  
Cfm3ExportPubKey returned: 0x00 : HSM Return: SUCCESS
```

Excluir chaves com o KMU e o CMU

Use o comando [deleteKey](#) para excluir uma chave, conforme o exemplo a seguir. Somente o proprietário da chave pode excluí-la.

```
Command: deleteKey -k 524300  
Cfm3DeleteKey returned: 0x00 : HSM Return: SUCCESS  
  
Cluster Error Status  
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS  
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS  
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Compartilhar e descompartilhar chaves com o KMU e o CMU

Em AWS CloudHSM, a UC que cria a chave a possui. O proprietário gerencia a chave, pode exportá-la e excluí-la e pode usá-la em operações criptográficas. O proprietário também pode compartilhar a chave com outros usuários CU. Os usuários com quem a chave é compartilhada podem usá-la em operações criptográficas, mas não podem exportá-la ou excluí-la nem compartilhá-la com outros usuários.

Você pode compartilhar chaves com outros usuários da UC ao criar a chave, por exemplo, usando o `-u` parâmetro dos comandos [genSymKey](#) ou [gen RSAKey Pair](#). Para compartilhar chaves existentes com um usuário diferente do HSM, use a ferramenta de linha de comando [cloudhsm_mgmt_util](#). Isso é diferente da maioria das tarefas documentadas nesta seção, que usam a ferramenta de linha de comando [key_mgmt_util](#).

Antes de compartilhar uma chave, você deve iniciar o `cloudhsm_mgmt_util`, habilitar end-to-end a criptografia e fazer login no HSMs. Para conseguir compartilhar uma chave, faça login no HSM como o usuário de criptografia (CU) que tem a chave. Somente os proprietários de chaves podem compartilhar uma chave.

Use o `shareKey` comando para compartilhar ou cancelar o compartilhamento de uma chave, especificando o identificador da chave e o IDs do usuário ou usuários. Para compartilhar ou cancelar o compartilhamento com mais de um usuário, especifique uma lista de usuários separada por vírgula. IDs Para compartilhar uma chave, use 1 como o último parâmetro do comando, como no exemplo a seguir. Para descompartilhar, use 0.

```
aws-cloudhsm > shareKey 524295 4 1
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)? y
shareKey success on server 0(10.0.2.9)
shareKey success on server 1(10.0.3.11)
shareKey success on server 2(10.0.1.12)
```

Veja a seguir a sintaxe do comando `shareKey`.

```
aws-cloudhsm > shareKey <key handle> <user ID> <Boolean: 1 for share, 0 for unshare>
```

Como marcar uma chave como confiável com o Utilitário AWS CloudHSM de Gerenciamento

O conteúdo desta seção fornece instruções sobre como usar o utilitário AWS CloudHSM de gerenciamento (CMU) para marcar uma chave como confiável.

1. Usando o comando [LoginHSM](#), faça login do como responsável pela criptografia (CO).
2. Use o comando [Defina os atributos das AWS CloudHSM chaves usando CMU](#) com `OBJ_ATTR_TRUSTED` (valor 134) definido como `true` (1).

```
aws-cloudhsm > setAttribute <Key Handle> 134 1
```

Backups de cluster em AWS CloudHSM

AWS CloudHSM faz backups periódicos do seu cluster pelo menos uma vez a cada 24 horas. Cada backup contém cópias criptografadas dos seguintes dados:

- Usuários (COs CUs, e AUs)
- Material e certificados de chave
- Configuração e políticas do módulo de segurança de hardware (HSM)

Você não pode instruir o serviço a fazer backups, mas pode realizar determinadas ações que forçam o serviço a criar um backup. O serviço faz um backup quando você executa qualquer uma das seguintes ações:

- Ativar um cluster
- Adicionar um HSM a um cluster ativo
- Remover um HSM de um cluster ativo

AWS CloudHSM exclui backups com base na política de retenção de backup que você definiu ao criar clusters. Para obter informações sobre como gerenciar a política de retenção de backup, consulte [Configurar a retenção de backup](#).

Tópicos

- [Trabalhando com backups AWS CloudHSM de cluster](#)
- [Excluir backups AWS CloudHSM do cluster](#)
- [Restaurar AWS CloudHSM backups](#)
- [Configurar a política de retenção de AWS CloudHSM backup](#)
- [Copiar backups de AWS CloudHSM cluster entre AWS regiões](#)
- [Trabalhando com backups compartilhados em AWS CloudHSM](#)

Trabalhando com backups AWS CloudHSM de cluster

Quando você adiciona um módulo de segurança de hardware (HSM) a um cluster AWS CloudHSM que anteriormente continha um ou mais ativos HSMs, o serviço restaura o backup mais recente no novo HSM. Use backups para gerenciar seu HSMs uso com pouca frequência. Quando não precisar

do HSM, exclua o HSM para acionar um backup. Posteriormente, quando precisar do HSM, crie um novo no mesmo cluster e essa ação restaurará o backup que você criou anteriormente com a operação de exclusão do HSM.

Remover chaves expiradas ou usuários inativos

Pode ser útil remover materiais de criptografia indesejados de seu ambiente, como uma chave expirada ou um usuário inativo. Este é um processo em duas etapas. Primeiro, exclua esses materiais do seu HSM. Em seguida, exclua todos os backups existentes. Seguir esse processo garante que você não restaure as informações excluídas ao inicializar um novo cluster a partir do backup. Para obter mais informações, consulte [the section called “Excluir backups”](#).

Considerar a recuperação de desastres

Você pode criar um cluster a partir de um backup. Talvez você queira fazer isso para definir um ponto de recuperação para seu cluster. Indique um backup que contenha todos os usuários, material da chave e certificados que você deseja em seu ponto de recuperação e, em seguida, use esse backup para criar um novo cluster. Para obter mais informações sobre como criar um cluster de um backup, consulte [Criar clusters de backups](#).

Você também pode copiar um backup de um cluster para uma região diferente, em que ele pode ser usado para criar outro cluster como um clone do original. Você pode fazer isso por vários motivos, incluindo simplificar o processo de recuperação de desastres. Para obter mais informações sobre copiar um backup entre regiões, consulte [Copiar um backup entre regiões](#).

Excluir backups AWS CloudHSM do cluster

Depois de excluir um backup de AWS CloudHSM cluster, o serviço retém o backup por sete dias, período durante o qual você pode restaurar o backup. Após o período de sete dias, você não poderá mais restaurar o backup. Para obter mais informações sobre gerenciar backups, consulte [Backups do cluster](#).

A tabela a seguir descreve como excluir um backup.

Console

Para excluir um backup (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.

2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. No painel de navegação, selecione Backups.
4. Escolha um backup para excluir.
5. Para excluir o backup selecionado, escolha Ações, Excluir.

A caixa de diálogo Excluir backups é exibida.

6. Escolha Excluir.

O estado do backup muda para PENDING_DELETE. Você pode restaurar um backup que está pendente de exclusão por até 7 dias após solicitar a exclusão.

Para listar backups (AWS CLI)

- Se quiser ver uma lista de todos os backups com o status PENDING_DELETION, execute o comando describe-backups e inclua states=PENDING_DELETION como filtro.

```
$ aws cloudhsmv2 describe-backups --filters states=PENDING_DELETION
{
  "Backups": [
    {
      "BackupId": "backup-ro5c4er4aac",
      "BackupState": "PENDING_DELETION",
      "ClusterId": "cluster-dygnwhmscg5",
      "CreateTimestamp": 1534461854.64,
      "DeleteTimestamp": 1536339805.522,
      "HsmType": "hsm2m.medium",
      "Mode": "NON_FIPS",
      "NeverExpires": false,
      "TagList": []
    }
  ]
}
```

AWS CLI

Verifique o status de um backup ou encontre sua ID usando o [describe-backups](#) comando do AWS CLI.

Para excluir um backup (AWS CLI)

- Em um prompt de comando, execute o comando [delete-backup](#), fornecendo o ID do backup a ser excluído.

```
$ aws cloudhsmv2 delete-backup --backup-id <backup ID>
{
  "Backup": {
    "CreateTimestamp": 1534461854.64,
    "ClusterId": "cluster-dygnwhmscg5",
    "BackupId": "backup-ro5c4er4aac",
    "BackupState": "PENDING_DELETION",
    "DeleteTimestamp": 1536339805.522,
    "HsmType": "hsm1.medium",
    "Mode": "FIPS"
  }
}
```

AWS CloudHSM API

Consulte [DeleteBackup](#) para saber como excluir backups usando a API.

Restaurar AWS CloudHSM backups

AWS CloudHSM mantém os backups excluídos por sete dias, período durante o qual você pode restaurar o backup. Após o período de sete dias, você não poderá mais restaurar o backup. Para obter mais informações sobre gerenciar backups, consulte [Backups do cluster](#).

A tabela a seguir descreve como excluir um backup.

Console

Para restaurar um backup (console)

- Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
- Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
- No painel de navegação, selecione Backups.
- Escolha um backup no PENDING_DELETE estado a ser restaurado.

5. Para restaurar o backup selecionado, escolha Ações, Restaurar.

AWS CLI

Para restaurar um backup (AWS CLI)

- Para restaurar um backup, execute o comando [restore-backup](#), fornecendo o ID do backup que tem o status PENDING_DELETION.

```
$ aws cloudhsmv2 restore-backup --backup-id <backup ID>
{
  "Backup": {
    "ClusterId": "cluster-dygnwhmscg5",
    "CreateTimestamp": 1534461854.64,
    "BackupState": "READY",
    "BackupId": "backup-ro5c4er4aac"
  }
}
```

AWS CloudHSM API

Consulte [RestoreBackup](#) para saber como restaurar backups usando a API.

Configurar a política de retenção de AWS CloudHSM backup

AWS CloudHSM limpa os backups com base na política de retenção de backup que você definiu ao criar um cluster. A política de retenção de backup se aplica aos clusters. Se você mover um backup para uma região diferente, esse backup não estará mais associado a um cluster e não terá política de retenção de backup. Você deve excluir manualmente todos os backups não associados a um cluster. AWS CloudHSM não exclui o último backup do cluster. Para clusters migrados do hsm1.medium, AWS CloudHSM retém o backup hsm1.medium mais recente até que o hsm1.medium chegue ao fim da vida útil.

[AWS CloudTrail](#) relata backups marcados para exclusão. Você pode restaurar os backups que o serviço limpa da mesma forma que restauraria os [backups excluídos manualmente](#). Para evitar uma sobreposição de comandos, você deve alterar a política de retenção de backup do cluster antes de restaurar um backup excluído pelo serviço. Se quiser manter a mesma política de retenção e

preservar alguns backups, você pode especificar que o serviço [excluir os backups](#) da política de retenção de backup em cluster.

Para obter mais informações sobre AWS CloudHSM preços, consulte [Reduza os custos escalando de acordo com suas necessidades](#).

Retenção de backup gerenciada

Os clusters criados antes de 18 de novembro de 2020 têm uma política de retenção de backup de 90 dias mais a idade do cluster. Por exemplo, se você criasse um cluster em 18 de novembro de 2019, o serviço atribuiria ao seu cluster uma política de retenção de backup de um ano mais 90 dias (455 dias). Você pode definir esse período para qualquer número entre 7 e 379 dias. AWS CloudHSM não exclui o último backup do cluster. Para obter mais informações sobre gerenciar backups, consulte [Backups do cluster](#).

Note

Você pode desativar totalmente a retenção gerenciada de backup entrando em contato com [AWS Support](#).

A tabela a seguir descreve como definir a retenção de backups.

Console

Para configurar a retenção de backup (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. Clique no ID do cluster de um cluster no estado ativo para gerenciar a política de retenção de backup desse cluster.
4. Para alterar a política de retenção de backup, escolha Ações, Alterar período de retenção de backup.

A caixa de diálogo Alterar período de retenção do backup é exibida.

5. Em Período de retenção de backup (em dias), digite um valor entre 7 e 379 dias.
6. Escolha Alterar período de retenção de backup.

Para excluir ou incluir um backup da política de retenção de backup (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para visualizar seus backups, escolha Backups no painel de navegação.
3. Clique no ID de backup de um backup no estado pronto para excluir ou incluir.
4. Na página Detalhes do backup, execute uma das seguintes ações.
 - Para excluir um backup com uma data no Prazo de expiração, escolha Ações, Desativar expiração.
 - Para incluir um backup que não expire, escolha Ações, Usar política de retenção de cluster.

AWS CLI

Para configurar a política de retenção de backup (AWS CLI)

- Em um prompt de comando, emita o comando `modify-cluster`. Especifique o ID do cluster e a política de retenção de backup.

```
$ aws cloudhsmv2 modify-cluster --cluster-id <cluster ID> \
                                --backup-retention-policy
                                Type=DAYS,Value=<number of days to retain backups>
{
  "Cluster": {
    "BackupPolicy": "DEFAULT",
    "BackupRetentionPolicy": {
      "Type": "DAYS",
      "Value": 90
    },
  },
  "Certificates": {},
  "ClusterId": "cluster-kdmrayrc7gi",
  "CreateTimestamp": 1504903546.035,
  "Hsms": [],
  "HsmType": "hsm1.medium",
  "SecurityGroup": "sg-40399d28",
  "State": "ACTIVE",
  "SubnetMapping": {
    "us-east-2a": "subnet-f1d6e798",
    "us-east-2c": "subnet-0e358c43",
  },
}
```

```

        "us-east-2b": "subnet-40ed9d3b"
    },
    "TagList": [
        {
            "Key": "Cost Center",
            "Value": "12345"
        }
    ],
    "VpcId": "vpc-641d3c0d"
}
}

```

Para excluir um backup da política de retenção de backup (AWS CLI)

- Em um prompt de comando, emita o comando `modify-backup-attributes`. Especifique o ID do backup e defina o sinalizador permanente para preservar o backup.

```

$ aws cloudhsmv2 modify-backup-attributes --backup-id <backup ID> \
                                         --never-expires
{
  "Backup": {
    "BackupId": "backup-ro5c4er4aac",
    "BackupState": "READY",
    "ClusterId": "cluster-dygnwhmscg5",
    "NeverExpires": true
  }
}

```

Para incluir um backup na política de retenção de backup (AWS CLI)

- Em um prompt de comando, emita o comando `modify-backup-attributes`. Especifique a ID do backup e defina o `no-never-expires` sinalizador para incluir o backup na política de retenção de backup, o que significa que o serviço acabará por excluir o backup.

```

$ aws cloudhsmv2 modify-backup-attributes --backup-id <backup ID> \
                                         --no-never-expires
{
  "Backup": {
    "BackupId": "backup-ro5c4er4aac",
    "BackupState": "READY",

```

```
"ClusterId": "cluster-dygnwhmscg5",  
"NeverExpires": false  
}  
}
```

AWS CloudHSM API

Consulte os tópicos a seguir para saber como gerenciar a retenção de backups usando a API.

- [ModifyCluster](#)
- [ModifyBackupAttributes](#)

Copiar backups de AWS CloudHSM cluster entre AWS regiões

[Você pode copiar backups de AWS CloudHSM cluster entre regiões por vários motivos, incluindo resiliência entre regiões, cargas de trabalho globais e recuperação de desastres.](#) Depois de copiar os backups, eles aparecem na região de destino com um status CREATE_IN_PROGRESS. Após a conclusão bem-sucedida da cópia, o status do backup copiado será READY. Se houve falha em uma cópia, o status do backup será alterado para DELETED. Verifique os parâmetros de entrada no caso de erros e verifique se o backup de origem especificado não está em um estado DELETED antes de executar novamente a operação. Para obter informações sobre como criar um cluster de um backup, consulte [Backups do cluster](#) ou [Criar clusters de backups](#).

Observe o seguinte:

- Para copiar um backup de cluster para uma região de destino, sua conta deve ter as permissões de política do IAM corretas. Para copiar o backup para outra região, sua política do IAM deve permitir o acesso à região de origem em que o backup está localizado. Após copiado entre as regiões, sua política do IAM deverá permitir o acesso à região de destino para interagir com o backup copiado, que inclui o uso da operação [CreateCluster](#). Para obter mais informações, consulte [Criar administradores do IAM](#).
- O cluster original e o cluster que pode ser criado de um backup na região de destino não estão vinculados. Você deve gerenciar cada um desses clusters de forma independente. Para obter mais informações, consulte [Clusters](#).
- Os backups não podem ser copiados entre AWS partições. Os backups podem ser copiados entre regiões AWS restritas e regiões padrão dentro da mesma partição.

Copiar backups para diferentes regiões (console)

Copiar backups para diferentes regiões (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. No painel de navegação, selecione Backups.
4. Para copiar um backup de cluster em outra região.
5. Para copiar o backup selecionado, escolha Ações, Copiar backup para outra região.

A caixa de diálogo Copiar backup para outra região é exibida.

6. Em Região de destino, escolha uma região em Selecionar uma região.
7. (Opcional) Digite uma chave de tag e um valor de tag opcional. Para adicionar mais de uma tag ao cluster, selecione Adicionar tag.
8. Selecione Copy backup (Copiar backup).

Copiar backups para diferentes regiões (AWS CLI)

Para determinar o ID de backup, execute o comando [describe-backups](#).

Copiar backups para diferentes regiões (AWS CLI)

- Em um prompt de comando, execute o comando [copy-backup-to-region](#). Especifique a região de destino e o ID do backup de origem. Se você especificar um ID de backup, o backup associado será copiado.

```
$ aws cloudhsmv2 copy-backup-to-region --destination-region <destination region> \
                                     --backup-id <backup ID>
```

Copiar backups para diferentes regiões (AWS CloudHSM API)

Consulte o tópico a seguir para saber como copiar backups para diferentes regiões usando a API.

- [CopyBackupToRegion](#)

Trabalhando com backups compartilhados em AWS CloudHSM

O CloudHSM se integra AWS Resource Access Manager com AWS RAM() para permitir o compartilhamento de recursos. AWS RAM é um serviço que permite que você compartilhe alguns recursos do CloudHSM com Contas da AWS outras pessoas ou por meio dela. AWS Organizations Com AWS RAM, você compartilha recursos de sua propriedade criando um compartilhamento de recursos. Um compartilhamento de atributos especifica os atributos a serem compartilhados, e os consumidores com os quais compartilhá-los. Os consumidores podem incluir:

- Específico Contas da AWS dentro ou fora de sua organização em AWS Organizations
- Uma unidade organizacional dentro de sua organização em AWS Organizations
- Uma organização inteira em AWS Organizations

Para obter mais informações sobre AWS RAM, consulte o [Guia AWS RAM do usuário](#).

Este tópico explica como compartilhar recursos que você possui e como usar os recursos que são compartilhados com você.

Conteúdo

- [Pré-requisitos para compartilhar backups](#)
- [Compartilhar um backup](#)
- [Cancelar o compartilhamento de um backup](#)
- [Identificar um backup compartilhado](#)
- [Permissões para backups compartilhados](#)
- [Faturamento e medição](#)

Pré-requisitos para compartilhar backups

- Para compartilhar um backup, você deve possuí-lo em seu Conta da AWS. Isso significa que o recurso deve ser alocado ou provisionado em sua conta. Não é possível compartilhar um backup que tenha sido compartilhado com você.
- Para compartilhar um backup, ele deve estar no estado READY.
- Para compartilhar uma malha com a sua organização ou com uma unidade organizacional no AWS Organizations, é necessário habilitar o compartilhamento com o AWS Organizations. Para obter

mais informações, consulte [Habilitar o compartilhamento com o AWS Organizations](#) no Guia do usuário do AWS RAM .

Compartilhar um backup

Ao compartilhar um backup com outras pessoas Contas da AWS, você permite que elas restaurem clusters do backup que contêm as chaves e os usuários armazenados no backup.

Para compartilhar um backup, é necessário adicioná-lo a um compartilhamento de recursos. Um compartilhamento de recursos é um recurso do AWS RAM que permite que você compartilhe seus recursos entre Contas da AWS. Um compartilhamento de recursos especifica os recursos a serem compartilhados, e os clientes com os quais compartilhá-los. Ao compartilhar um backup usando o console do CloudHSM, você a adiciona a um compartilhamento de recursos existente. Para adicionar o backup a um novo compartilhamento de recursos, primeiramente você deve criar o compartilhamento de recursos usando o [console do AWS RAM](#).

Se você faz parte de uma organização AWS Organizations e o compartilhamento dentro de sua organização está ativado, os consumidores em sua organização recebem automaticamente acesso ao backup compartilhado. Caso contrário, os consumidores receberão um convite para participar do compartilhamento de recursos e acesso ao backup compartilhado depois de aceitar o convite.

Você pode compartilhar um backup de sua propriedade usando o AWS RAM console ou AWS CLI.

Para compartilhar um backup que você possui usando o AWS RAM console

Consulte [Criar um compartilhamento de recursos](#) no Manual do usuário do AWS RAM .

Para compartilhar um backup que você possui (AWS RAM comando)

Use o comando [create-resource-share](#).

Para compartilhar um backup que você possui (comando do CloudHSM)

Important

Embora você possa compartilhar um backup usando a operação do PutResourcePolicy CloudHSM, recomendamos AWS Resource Access Manager usar AWS RAM() em vez disso. AWS RAM O uso oferece vários benefícios, pois cria a política para você, permite que vários recursos sejam compartilhados ao mesmo tempo e aumenta a capacidade de descoberta de recursos compartilhados. Se você usa PutResourcePolicy e deseja que os consumidores possam descrever os backups que você compartilhou com eles, você deve promover o

backup para um compartilhamento de AWS RAM recursos padrão usando a operação de AWS RAM `PromoteResourceShareCreatedFromPolicy` API.

Use o comando [put-resource-policy](#).

1. Crie um arquivo chamado `policy.json` e copie a política a seguir nele.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "111122223333"
      },
      "Action": [
        "cloudhsm:CreateCluster",
        "cloudhsm:DescribeBackups"
      ],
      "Resource": "arn:aws:cloudhsm:us-west-2:111122223333:backup/backup-to-  
share"
    }
  ]
}
```

2. Atualize `policy.json` com o ARN e os identificadores do backup e para compartilhá-lo. O exemplo a seguir concede acesso somente de leitura ao usuário raiz da AWS conta identificada por 123456789012.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": [
```

```

        "123456789012"
    ],
    },
    "Action": [
        "cloudhsm:CreateCluster",
        "cloudhsm:DescribeBackups"
    ],
    "Resource": "arn:aws:cloudhsm:us-west-2:123456789012:backup/backup-123"
}
]
}

```

Important

Você só pode conceder permissões DescribeBackups no nível da conta. Quando você compartilha um backup com outro cliente, qualquer diretor que tenha DescribeBackups permissão nessa conta pode descrever o backup.

3. Execute o comando [put-resource-policy](#).

```
$ aws cloudhsmv2 put-resource-policy --resource-arn <resource-arn> --policy file://policy.json
```

Note

Nesse ponto, o consumidor pode usar o backup, mas ele não aparecerá na DescribeBackups resposta com o parâmetro compartilhado. As próximas etapas descrevem como promover o compartilhamento de AWS RAM recursos para que o backup seja incluído na resposta.

4. Obtenha o ARN do compartilhamento de AWS RAM recursos.

```
$ aws ram list-resources --resource-owner SELF --resource-arns <backup-arn>
```

Isso retorna uma resposta semelhante a esta:

```
{
  "resources": [
```

```
{
  "arn": "<project-arn>",
  "type": "<type>",
  "resourceShareArn": "<resource-share-arn>",
  "creationTime": "<creation-time>",
  "lastUpdatedTime": "<last-update-time>"
}
```

Na resposta, copie o **<resource-share-arn>** valor a ser usado nas próximas etapas.

5. Execute o comando AWS RAM [promote-resource-share-created-from-policy](#).

```
$ aws ram promote-resource-share-created-from-policy --resource-share-arn <resource-share-arn>
```

6. Para validar se o compartilhamento de recursos foi promovido, você pode executar o AWS RAM [get-resource-shares](#) comando.

```
$ aws ram get-resource-shares --resource-owner SELF --resource-share-arns <resource-share-arn>
```

Quando a política é promovida, o `featureSet` listado na resposta é `STANDARD`. Isso também significa que o backup pode ser descrito pelas novas contas na política.

Cancelar o compartilhamento de um backup

Quando você cancela o compartilhamento de um recurso, o consumidor não pode mais usá-lo para restaurar um cluster. Os consumidores ainda poderão acessar todos os clusters que eles restauraram a partir do backup compartilhado.

Para cancelar o compartilhamento de um backup compartilhado de sua propriedade, é necessário removê-lo do compartilhamento de recursos. Você pode fazer isso usando o AWS RAM console ou AWS CLI.

Para cancelar o compartilhamento de um backup compartilhado que você possui usando o console AWS RAM

Consulte [Atualização de um compartilhamento de atributos](#) no Guia do usuário do AWS RAM .

Para cancelar o compartilhamento de um backup compartilhado que você possui (AWS RAM comando)

Use o comando [disassociate-resource-share](#).

Para cancelar o compartilhamento de um backup compartilhado que você possui (comando do CloudHSM)

Use o comando [delete-resource-policy](#).

```
$ aws cloudhsmv2 delete-resource-policy --resource-arn <resource-arn>
```

Identificar um backup compartilhado

Os consumidores podem identificar um backup compartilhado com eles usando o console e a AWS CLI do CloudHSM.

Para identificar backups compartilhados com você usando o console do CloudHSM

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar o Região da AWS, use o seletor de região no canto superior direito da página.
3. No painel de navegação, selecione Backups.
4. Na tabela, escolha a guia Backups compartilhados.

Para identificar backups compartilhados com você usando o AWS CLI

Use o comando [describe-backups](#) com o parâmetro `--shared` para retornar os backups compartilhados com você.

Permissões para backups compartilhados

Permissões para proprietários

Os proprietários do backup podem descrever e gerenciar um backup compartilhado, bem como usá-lo para restaurar um cluster.

Permissões para clientes

Os consumidores de backup não podem modificar um backup compartilhado, mas podem descrevê-lo e usá-lo para restaurar um cluster.

Faturamento e medição

Não há cobranças adicionais pelo compartilhamento de backups.

Clusters clonados em AWS CloudHSM

Use o Utilitário de AWS CloudHSM Gerenciamento (CMU) para sincronizar um cluster em uma região remota, se o cluster nessa região tiver sido criado originalmente a partir do backup de um cluster em outra região. Digamos que você copiou um cluster para outra região (destino) e depois queira sincronizar as alterações do cluster original (origem). Em cenários como esse, você usa a CMU para sincronizar os clusters. Você faz isso criando um novo arquivo de configuração CMU, especificando módulos de segurança de hardware (HSM) de ambos os clusters no novo arquivo e, em seguida, usando a CMU para se conectar ao cluster com esse arquivo.

Para usar a CMU em clusters clonados

1. Crie uma cópia do seu arquivo de configuração atual e altere o nome da cópia para outra coisa.

Por exemplo, use os seguintes locais de arquivo para localizar e criar uma cópia do seu arquivo de configuração atual e, em seguida, altere o nome da cópia de `cloudhsm_mgmt_config.cfg` para `syncConfig.cfg`.

- Linux: `/opt/cloudhsm/etc/cloudhsm_mgmt_config.cfg`
- Windows: `C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_config.cfg`

2. Na cópia renomeada, adicione o IP da interface de rede elástica (ENI) do HSM de destino (o HSM na região externa que precisa ser sincronizado). Recomendamos que você adicione o HSM de destino abaixo do HSM de origem.

```
{
  ...
  "servers": [
    {
      ...
      "hostname": "<ENI Source IP>",
      ...
    },
    {
      ...
      "hostname": "<ENI Destination IP>",
      ...
    }
  ]
}
```


Para obter mais informações sobre endereço IP, consulte [the section called “Obter um endereço IP para um HSM”](#).

3. Inicialize o CMU com o novo arquivo de configuração:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/userSync.cfg
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\userSync.cfg
```

4. Verifique as mensagens de status retornadas para garantir que a CMU esteja conectada a todas as opções desejadas HSMs e determine qual das ENI retornadas IPs corresponde a cada cluster. Use SyncUser e SyncKey para sincronizar manualmente usuários e chaves. Para obter mais informações, consulte [syncUser](#) e [syncKey](#).

Obter um endereço IP para um HSM

Use esta seção para obter um endereço IP para um HSM.

Para obter um endereço IP para um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. Para abrir a página de detalhes do cluster, na tabela do cluster, escolha o ID do cluster.
4. Para obter o endereço IP, vá até a HSMs guia. Para IPv4 clusters, escolha um endereço listado em IPv4 Endereço ENI. Para clusters de pilha dupla, use o endereço ENI IPv4 ou ENI. IPv6

Para obter um endereço IP para um HSM (AWS CLI)

- Obtenha o endereço IP de um HSM usando o comando [describe-clusters](#) do AWS CLI. Na saída do comando, o endereço IP dos HSMs são os valores de `EniIp` e `EniIpV6` (se for um cluster de pilha dupla).

```
$ aws cloudhsmv2 describe-clusters
{
  "Clusters": [
    { ... }
    "Hsms": [
      {
...
        "EniIp": "10.0.0.9",
...
      },
      {
...
        "EniIp": "10.0.1.6",
        "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
    }
  ]
}
```

Tópicos relacionados

- [syncUser](#)
- [syncKey](#)
- [Copiar um backup entre regiões](#)

AWS CloudHSM Recursos de tags

Uma tag é um rótulo que você atribui a um AWS recurso. É possível atribuir tags a clusters do AWS CloudHSM . Cada tag consiste em uma chave da tag e um valor da tag, ambos definidos por você. Por exemplo, a chave da tag pode ser Central de custos e o valor da tag pode ser 12345. As chaves de tag devem ser exclusivas para cada cluster.

As tags podem ser usadas para diversas finalidades. Um uso comum é para categorizar e monitorar os custos da AWS . É possível aplicar tags que representem categorias de negócios (como centros de custos, nomes da aplicações ou proprietários) para organizar seus custos de vários serviços. 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 relatório para visualizar seus AWS CloudHSM custos em termos de projetos ou aplicativos, em vez de visualizar todos os AWS CloudHSM custos como um único item de linha.

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 .

Você pode usar o [AWS CloudHSM console](#) ou uma das [ferramentas da AWS SDKs linha de comando](#) para adicionar, atualizar, listar e remover tags.

Tópicos

- [Adicionar ou atualizar tags para AWS CloudHSM recursos](#)
- [Listar tags para AWS CloudHSM recursos](#)
- [Remover tags dos AWS CloudHSM recursos](#)

Adicionar ou atualizar tags para AWS CloudHSM recursos

Você pode adicionar ou atualizar tags no [console do AWS CloudHSM](#), na [AWS Command Line Interface \(AWS CLI\)](#) ou na API do AWS CloudHSM .

Para adicionar ou atualizar tags (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Escolha o cluster que você está marcando.
3. Escolha Tags.

4. Para adicionar uma tag, faça o seguinte:
 - a. Escolha Edit Tag (Editar tag) e Add Tag (Adicionar tag).
 - b. Em Key (Chave), digite uma chave para a tag.
 - c. (Opcional) Em Value (Valor), digite um valor para a tag.
 - d. Escolha Salvar.
5. Para atualizar uma tag, faça o seguinte:
 - a. Escolha Edit Tag (Editar tag).

 Note

Se você atualizar a chave de uma tag existente, o console excluirá a tag existente e criará uma nova.

- b. Digite o novo valor da tag.
- c. Escolha Salvar.

Para adicionar ou atualizar tags (AWS CLI)

1. No prompt de comando, emita o comando [tag-resource](#), especificando as tags e o ID do cluster que você está marcando. Se você não conhece o ID do cluster, emita o comando [describe-clusters](#).

```
$ aws cloudhsmv2 tag-resource --resource-id <cluster ID> \  
--tag-list Key="<tag key>",Value="<tag value>"
```

2. Para atualizar tags, use o mesmo comando, mas especifique uma chave de tag existente. Quando você especifica um novo valor de tag para uma tag existente, esta é substituída pelo novo valor.

Para adicionar ou atualizar tags (AWS CloudHSM API)

- Envie uma solicitação [TagResource](#). Especifique as tags e o ID do cluster que você está marcando.

Listar tags para AWS CloudHSM recursos

Você pode listar as tags de um cluster a partir do [AWS CloudHSM console](#) [AWS CLI](#), do ou da AWS CloudHSM API.

Para listar tags (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Escolha o cluster cujas tags você está listando.
3. Escolha Tags.

Para listar tags (AWS CLI)

- No prompt de comando, emita o comando [list-tags](#), especificando o ID do cluster cujas tags você está listando. Se você não conhece o ID do cluster, emita o comando [describe-clusters](#).

```
$ aws cloudhsmv2 list-tags --resource-id <cluster ID>
{
  "TagList": [
    {
      "Key": "Cost Center",
      "Value": "12345"
    }
  ]
}
```

Para listar tags (AWS CloudHSM API)

- Envie uma solicitação [ListTags](#), especificando o ID do cluster cujas tags você está listando.

Remover tags dos AWS CloudHSM recursos

Você pode remover tags de um AWS CloudHSM cluster usando o [AWS CloudHSM console](#) [AWS CLI](#), o ou a AWS CloudHSM API.

Para remover tags (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.

2. Escolha o cluster cujas tags você está removendo.
3. Escolha Tags.
4. Escolha Edit Tag (Editar tag) e depois Remove tag (Remover tag) para a tag que você deseja remover.
5. Escolha Salvar.

Para remover tags (AWS CLI)

- No prompt de comando, emita o comando [untag-resource](#), especificando as chaves da tag que você está removendo e o ID do cluster cujas tags você está removendo. Ao usar o AWS CLI para remover tags, especifique somente as chaves da tag, não os valores da tag.

```
$ aws cloudhsmv2 untag-resource --resource-id <cluster ID> \  
                                --tag-key-list "<tag key>"
```

Para remover tags (AWS CloudHSM API)

- Envie uma [UntagResources](#) solicitação na AWS CloudHSM API, especificando o ID do cluster e as tags que você está removendo.

AWS CloudHSM ferramentas de linha de comando

Além do AWS Command Line Interface (AWS CLI) que você usa para gerenciar seus recursos da AWS, AWS CloudHSM oferece ferramentas de linha de comando para criar e gerenciar usuários e chaves do módulo de segurança de hardware (HSM) em seu. HSMs Em AWS CloudHSM, você usa a CLI familiar para gerenciar seu cluster e as ferramentas de linha de comando do CloudHSM para gerenciar seu HSM.

Estas são as várias ferramentas de linha de comando:

Para gerenciar HSMs e agrupar

[HSMv2 Comandos do Cloud AWS CLI](#) e [HSM2 PowerShell cmdlets no módulo Shell AWSPower](#)

- Essas ferramentas obtêm, criam, excluem e marcam AWS CloudHSM clusters e HSMs:
- [Para usar os comandos nos HSMv2 comandos do Cloud na CLI, você precisa instalar e configurar.](#) AWS CLI
- HSM2 PowerShell Os [cmdlets no módulo AWSPower Shell](#) estão disponíveis em um PowerShell módulo Windows e em um módulo Core multiplataforma PowerShell .

Como gerenciar usuários do HSM.

[CLI do CloudHSM](#)

- Use a [CloudHSM CLI](#) para criar usuários, excluir usuários, listar usuários, alterar senhas de usuários e atualizar a autenticação multifator (MFA) do usuário. Ele não está incluído no software cliente do AWS CloudHSM. Para obter orientação sobre a instalação dessa ferramenta, consulte [Instalar e configurar a CLI do CloudHSM](#).

Ferramentas auxiliares

Duas ferramentas ajudam você a usar AWS CloudHSM ferramentas e bibliotecas de software:

- A [ferramenta de configuração](#) atualiza seus arquivos de configuração do cliente CloudHSM. Isso permite AWS CloudHSM sincronizar o HSMs em um cluster.

AWS CloudHSM oferece duas versões principais, e o Client SDK 5 é a mais recente. Ele oferece uma variedade de vantagens em relação ao Client SDK 3 (a série anterior).

- [pkpspeed](#) mede o desempenho do seu hardware do HSM independente das bibliotecas de software.

Ferramentas para o anterior SDKs

Use a ferramenta de gerenciamento de chaves (KMU) para criar, excluir, importar e exportar chaves simétricas e pares de chaves assimétricas:

- [key_mgmt_util](#). Essa ferramenta está incluída no software cliente do AWS CloudHSM .

Use a ferramenta de gerenciamento CloudHSM (CMU) para criar e excluir usuários do HSM, incluindo a implementação da autenticação de quórum das tarefas de gerenciamento de usuários

- [cloudhsm_mgmt_util](#). Essa ferramenta está incluída no software cliente do AWS CloudHSM .

Os tópicos a seguir descrevem com mais detalhes as ferramentas de linha de comando disponíveis para gerenciar e usar o . AWS CloudHSM

Tópicos

- [AWS CloudHSM ferramenta de configuração](#)
- [AWS CloudHSM Interface de linha de comando \(CLI\)](#)
- [AWS CloudHSM Utilitário de gerenciamento \(CMU\)](#)
- [AWS CloudHSM Utilitário de gerenciamento de chaves \(KMU\)](#)

AWS CloudHSM ferramenta de configuração

AWS CloudHSM sincroniza automaticamente os dados entre todos os módulos de segurança de hardware (HSM) em um cluster. A ferramenta configure atualiza os dados do HSM nos arquivos de configuração que são usados pelos mecanismos de sincronização. Use configure para atualizar os dados do HSM antes de usar as ferramentas de linha de comando, especialmente quando as do cluster forem HSMs alteradas.

AWS CloudHSM inclui duas versões principais do SDK do cliente:

- Client SDK 5: este é o nosso SDK de cliente padrão e mais recente. Para obter informações sobre os benefícios e vantagens que ela oferece, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).
- Client SDK 3: este é o nosso SDK do cliente mais antigo. Ele inclui um conjunto completo de componentes para ferramentas de gerenciamento e compatibilidade de aplicativos baseados em plataformas e linguagens.

Para obter instruções sobre como migrar do Client SDK 3 para o Client SDK 5, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Tópicos

- [AWS CloudHSM Ferramenta de configuração do Client SDK 5](#)
- [AWS CloudHSM Ferramenta de configuração do Client SDK 3](#)

AWS CloudHSM Ferramenta de configuração do Client SDK 5

Use a ferramenta de configuração AWS CloudHSM do Client SDK 5 para atualizar os arquivos de configuração do lado do cliente.

Cada componente no Client SDK 5 inclui uma ferramenta de configuração com um designador do componente no nome do arquivo da ferramenta de configuração. Por exemplo, a biblioteca PKCS #11 do Client SDK 5 inclui uma ferramenta de configuração chamada `configure-pkcs11` no Linux ou `configure-pkcs11.exe` no Windows.

Tópicos

- [AWS CloudHSM Sintaxe de configuração do SDK 5 do cliente](#)
- [AWS CloudHSM Parâmetros de configuração do SDK do cliente 5](#)
- [AWS CloudHSM Exemplos de configuração do Client SDK 5](#)
- [Provedor OpenSSL Bootstrap](#)
- [Configurações avançadas para a ferramenta de configuração do Client SDK 5](#)
- [AWS CloudHSM Tópicos relacionados ao Client SDK 5](#)

AWS CloudHSM Sintaxe de configuração do SDK 5 do cliente

A tabela a seguir ilustra a sintaxe dos arquivos de AWS CloudHSM configuração do Client SDK 5. Para obter mais informações sobre os parâmetros, consulte [the section called “Parâmetros”](#).

PKCS #11

```
Usage: configure-pkcs11[ .exe ] [OPTIONS]
```

Options:

```
--disable-certificate-storage
    Disables Certificate Storage
```

```

--enable-certificate-storage
    Enables Certificate Storage
-a <HSM ENI IP>...
    The address of the HSM instance
--cluster-id <CLUSTER ID>
    The id of the cluster containing the HSM instance(s)
--disable-key-availability-check
    Disables key availability check during key use
--enable-key-availability-check
    Enables key availability check during key use
--disable-validate-key-at-init
    Disables parameter validation during initialization of crypto operations
--enable-validate-key-at-init
    Enables parameter validation during initialization of crypto operations
--endpoint <ENDPOINT>
    Specify the AWS CloudHSM API Endpoint
--region <REGION>
    The region of the cluster
--hsm-ca-cert <HSM CA CERTIFICATE FILE>
    The HSM CA certificate file
--log-type <LOG TYPE>
    The log type [possible values: term, file]
--log-file <LOG FILE>
    The log file
--log-level <LOG LEVEL>
    The logging level [possible values: error, warn, info, debug, trace]
--log-rotation <LOG ROTATION>
    The log rotation interval [possible values: never, hourly, daily]
--default-retry-mode <RETRY MODE>
    The default method of retry to use for certain non-terminal failures
[possible values: off, standard]
--client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
    The client certificate used for TLS client-hsm mutual authentication
--client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
    The client private key used for TLS client-hsm mutual authentication
-h, --help
    Print help

```

OpenSSL

Usage: configure-dyn[.exe] [OPTIONS]

Options:

```

-a <HSM ENI IP>...
    The address of the HSM instance
--cluster-id <CLUSTER ID>
    The id of the cluster containing the HSM instance(s)
--disable-key-availability-check
    Disables key availability check during key use
--enable-key-availability-check
    Enables key availability check during key use
--disable-validate-key-at-init
    Disables parameter validation during initialization of crypto operations
--enable-validate-key-at-init
    Enables parameter validation during initialization of crypto operations
--endpoint <ENDPOINT>
    Specify the AWS CloudHSM API Endpoint
--region <REGION>
    The region of the cluster
--hsm-ca-cert <HSM CA CERTIFICATE FILE>
    The HSM CA certificate file
--log-type <LOG TYPE>
    The log type [possible values: term, file]
--log-file <LOG FILE>
    The log file
--log-level <LOG LEVEL>
    The logging level [possible values: error, warn, info, debug, trace]
--log-rotation <LOG ROTATION>
    The log rotation interval [possible values: never, hourly, daily]
--default-retry-mode <RETRY MODE>
    The default method of retry to use for certain non-terminal failures
[possible values: off, standard]
--client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
    The client certificate used for TLS client-hsm mutual authentication
--client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
    The client private key used for TLS client-hsm mutual authentication
-h, --help
    Print help

```

KSP

Usage: configure-ksp.exe [OPTIONS]

Options:

```

-a <HSM ENI IP>...
    The address of the HSM instance

```

```
--server-client-cert-file <CLIENT CERTIFICATE FILE>
    The client certificate used for TLS client-server mutual authentication
--server-client-key-file <CLIENT KEY FILE>
    The client private key used for TLS client-server mutual authentication
--cluster-id <CLUSTER ID>
    The id of the cluster containing the HSM instance(s)
--disable-key-availability-check
    Disables key availability check during key use
--enable-key-availability-check
    Enables key availability check during key use
--disable-validate-key-at-init
    Disables parameter validation during initialization of crypto operations
--enable-validate-key-at-init
    Enables parameter validation during initialization of crypto operations
--endpoint <ENDPOINT>
    Specify the AWS CloudHSM API Endpoint
--region <REGION>
    The region of the cluster
--hsm-ca-cert <HSM CA CERTIFICATE FILE>
    The HSM CA certificate file
--log-type <LOG TYPE>
    The log type [possible values: term, file]
--log-file <LOG FILE>
    The log file
--log-level <LOG LEVEL>
    The logging level [possible values: error, warn, info, debug, trace]
--log-rotation <LOG ROTATION>
    The log rotation interval [possible values: never, hourly, daily]
--default-retry-mode <RETRY MODE>
    The default method of retry to use for certain non-terminal failures
[possible values: off, standard]
--client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
    The client certificate used for TLS client-hsm mutual authentication
--client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
    The client private key used for TLS client-hsm mutual authentication
--enable-sdk3-compatibility-mode
    Enables key file usage for KSP
--disable-sdk3-compatibility-mode
    Disables key file usage for KSP
-h, --help
    Print help
```

JCE

Usage: configure-jce[.exe] [OPTIONS]

Options:

-a <HSM ENI IP>...

The address of the HSM instance

--cluster-id <CLUSTER ID>

The id of the cluster containing the HSM instance(s)

--disable-key-availability-check

Disables key availability check during key use

--enable-key-availability-check

Enables key availability check during key use

--disable-validate-key-at-init

Disables parameter validation during initialization of crypto operations

--enable-validate-key-at-init

Enables parameter validation during initialization of crypto operations

--endpoint <ENDPOINT>

Specify the AWS CloudHSM API Endpoint

--region <REGION>

The region of the cluster

--hsm-ca-cert <HSM CA CERTIFICATE FILE>

The HSM CA certificate file

--log-type <LOG TYPE>

The log type [possible values: term, file]

--log-file <LOG FILE>

The log file

--log-level <LOG LEVEL>

The logging level [possible values: error, warn, info, debug, trace]

--log-rotation <LOG ROTATION>

The log rotation interval [possible values: never, hourly, daily]

--default-retry-mode <RETRY MODE>

The default method of retry to use for certain non-terminal failures
[possible values: off, standard]

--client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>

The client certificate used for TLS client-hsm mutual authentication

--client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>

The client private key used for TLS client-hsm mutual authentication

-h, --help

Print help

CloudHSM CLI

Usage: `configure-cli[ .exe ] [OPTIONS]`

Options:

`-a <HSM ENI IP>...`

The address of the HSM instance

`--cluster-id <CLUSTER ID>`

The id of the cluster containing the HSM instance(s)

`--disable-key-availability-check`

Disables key availability check during key use

`--enable-key-availability-check`

Enables key availability check during key use

`--disable-validate-key-at-init`

Disables parameter validation during initialization of crypto operations

`--enable-validate-key-at-init`

Enables parameter validation during initialization of crypto operations

`--endpoint <ENDPOINT>`

Specify the AWS CloudHSM API Endpoint

`--region <REGION>`

The region of the cluster

`--hsm-ca-cert <HSM CA CERTIFICATE FILE>`

The HSM CA certificate file

`--log-type <LOG TYPE>`

The log type [possible values: term, file]

`--log-file <LOG FILE>`

The log file

`--log-level <LOG LEVEL>`

The logging level [possible values: error, warn, info, debug, trace]

`--log-rotation <LOG ROTATION>`

The log rotation interval [possible values: never, hourly, daily]

`--default-retry-mode <RETRY MODE>`

The default method of retry to use for certain non-terminal failures
[possible values: off, standard]

`--client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>`

The client certificate used for TLS client-hsm mutual authentication

`--client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>`

The client private key used for TLS client-hsm mutual authentication

`-h, --help`

Print help

OpenSSL Provider

Usage: `configure-openssl-provider[ .exe ] [OPTIONS]`

Options:

`-a <HSM ENI IP>...`

The address of the HSM instance

`--cluster-id <CLUSTER ID>`

The id of the cluster containing the HSM instance(s)

`--disable-key-availability-check`

Disables key availability check during key use

`--enable-key-availability-check`

Enables key availability check during key use

`--disable-validate-key-at-init`

Disables parameter validation during initialization of crypto operations

`--enable-validate-key-at-init`

Enables parameter validation during initialization of crypto operations

`--endpoint <ENDPOINT>`

Specify the AWS CloudHSM API Endpoint

`--region <REGION>`

The region of the cluster

`--hsm-ca-cert <HSM CA CERTIFICATE FILE>`

The HSM CA certificate file

`--log-type <LOG TYPE>`

The log type [possible values: term, file]

`--log-file <LOG FILE>`

The log file

`--log-level <LOG LEVEL>`

The logging level [possible values: error, warn, info, debug, trace]

`--log-rotation <LOG ROTATION>`

The log rotation interval [possible values: never, hourly, daily]

`--default-retry-mode <RETRY MODE>`

The default method of retry to use for certain non-terminal failures
[possible values: off, standard]

`--client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>`

The client certificate used for TLS client-hsm mutual authentication

`--client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>`

The client private key used for TLS client-hsm mutual authentication

`-h, --help`

Print help

AWS CloudHSM Parâmetros de configuração do SDK do cliente 5

Veja a seguir uma lista de parâmetros para configurar o AWS CloudHSM Client SDK 5.

-uma **<ENI IP address>**

Adiciona o endereço IP especificado aos arquivos de configuração do SDK do cliente 5. Insira qualquer endereço IP ENI de um HSM do cluster. Para obter mais informações sobre como usar essa opção, consulte o [Bootstrap o Client SDK 5](#).

Obrigatório: Sim

--hsm-ca-cert **<customerCA certificate file path>**

Caminho para o diretório que armazena o certificado da autoridade de certificação (CA) usado para conectar instâncias do cliente EC2 ao cluster. Esse é o arquivo que você criou ao inicializar o cluster. Por padrão, o sistema procura esse arquivo no seguinte local:

Linux

```
/opt/cloudhsm/etc/customerCA.crt
```

Windows

```
C:\ProgramData\Amazon\CloudHSM\customerCA.crt
```

Para obter mais informações sobre como inicializar o cluster ou colocar o certificado, consulte [???](#) e [???](#).

Obrigatório: não

--ID do cluster **<cluster ID>**

Faz uma chamada `DescribeClusters` para encontrar todos os endereços IP da interface de rede elástica (ENI) do HSM no cluster associado ao ID do cluster. O sistema adiciona os endereços IP ENI aos arquivos de AWS CloudHSM configuração.

Note

Se você usar o `--cluster-id` parâmetro de uma instância do EC2 em uma VPC que não tem acesso à Internet pública, deverá criar uma interface VPC endpoint à qual se

conectar. AWS CloudHSM Para obter mais informações sobre os endpoints da VPC, consulte [???](#).

Obrigatório: não

--ponto final **<endpoint>**

Especifique o endpoint AWS CloudHSM da API usado para fazer a `DescribeClusters` chamada. Você deve definir essa opção em combinação com `--cluster-id`.

Obrigatório: não

--região **<region>**

Especifique a região do seu cluster. Você deve definir essa opção em combinação com `--cluster-id`.

Se você não fornecer o parâmetro `--region`, o sistema escolherá a região tentando ler as variáveis de ambiente `AWS_DEFAULT_REGION` ou `AWS_REGION`. Se essas variáveis não estiverem definidas, o sistema verificará a região associada ao seu perfil no seu arquivo de AWS Config (normalmente `~/.aws/config`), a menos que você tenha especificado um arquivo diferente na variável de ambiente `AWS_CONFIG_FILE`. Se nenhuma das opções acima for definida, o sistema usará a região `us-east-1` como padrão.

Obrigatório: não

--client-cert-hsm-tls -arquivo **<client certificate hsm tls path>**

Caminho para o certificado de cliente usado para autenticação mútua entre cliente TLS e HSM.

Use essa opção somente se você tiver registrado pelo menos uma âncora de confiança no HSM com a CLI do CloudHSM. Você deve definir essa opção em combinação com `--client-key-hsm-tls-file`.

Obrigatório: não

--client-key-hsm-tls -arquivo **<client key hsm tls path>**

Caminho para a chave do cliente usado para autenticação mútua entre cliente TLS e HSM.

Use essa opção somente se você tiver registrado pelo menos uma âncora de confiança no HSM com a CLI do CloudHSM. Você deve definir essa opção em combinação com `--client-cert-hsm-tls-file`.

Obrigatório: não

--nível de log **<error | warn | info | debug | trace>**

Especifica o nível mínimo de registro que o sistema deve gravar no arquivo de log. Cada nível inclui os níveis anteriores, com erro como nível mínimo e rastreie o nível máximo. Isso significa que, se você especificar erros, o sistema só gravará erros no log. Se você especificar rastreamento, o sistema gravará mensagens de erros, avisos, informações (info) e de depuração no log. Para obter mais informações, consulte [Atualização do Client SDK 5](#).

Obrigatório: não

--rotação de log **<daily | weekly>**

Especifica a frequência com que o sistema gira os logs. Para obter mais informações, consulte [Atualização do Client SDK 5](#).

Obrigatório: não

--arquivo de log **<file name with path>**

Especifica onde o sistema gravará o arquivo de log. Para obter mais informações, consulte [Atualização do Client SDK 5](#).

Obrigatório: não

--tipo de log **<term | file>**

Especifica se o sistema gravará o log em um arquivo ou terminal. Para obter mais informações, consulte [Atualização do Client SDK 5](#).

Obrigatório: não

-h | --help

Exibe ajuda.

Obrigatório: não

--disable-key-availability-check

Sinalize para desativar o quórum de disponibilidade de chaves. Use esse sinalizador para indicar se AWS CloudHSM deve desativar o quórum de disponibilidade de chaves e você pode usar chaves que existem em apenas um HSM no cluster. Para obter mais informações sobre como usar esse sinalizador para definir o quórum de disponibilidade de chaves, consulte [???](#).

Obrigatório: não

`--enable-key-availability-check`

Sinalize para ativar o quórum de disponibilidade de chaves. Use esse sinalizador para indicar que você AWS CloudHSM deve usar o quorum de disponibilidade de chaves e não permitir que você use chaves até que essas chaves existam em duas HSMs no cluster. Para obter mais informações sobre como usar esse sinalizador para definir o quórum de disponibilidade de chaves, consulte [???](#).

Habilitada por padrão.

Obrigatório: não

`-- disable-validate-key-at -init`

Melhora o desempenho especificando que você pode pular uma chamada de inicialização para verificar as permissões em uma chave para chamadas subsequentes. Use com cautela.

Antecedentes: Alguns mecanismos na biblioteca PKCS #11 oferecem suporte a operações de várias partes em que uma chamada de inicialização verifica se você pode usar a chave para chamadas subsequentes. Isso requer uma chamada de verificação para o HSM, o que adiciona latência à operação geral. Essa opção permite que você desative a chamada subsequente e potencialmente melhore o desempenho.

Obrigatório: não

`-- enable-validate-key-at -init`

Especifica que você deve usar uma chamada de inicialização para verificar as permissões em uma chave para chamadas subsequentes. Esta é a opção padrão. Use `enable-validate-key-at-init` para retomar essas chamadas de inicialização depois de usar `disable-validate-key-at-init` para suspendê-las.

Obrigatório: não

AWS CloudHSM Exemplos de configuração do Client SDK 5

Esses exemplos mostram como usar a ferramenta de configuração para o AWS CloudHSM Client SDK 5.

Bootstrap do Client SDK 5

Example

Este exemplo usa o parâmetro `-a` para atualizar os dados HSM do Client SDK 5. Para usar o `-a` parâmetro, você deve ter o endereço IP de um deles HSMs em seu cluster.

PKCS #11 library

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 -a <HSM IP addresses>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" -a <HSM IP addresses>
```

OpenSSL Dynamic Engine

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-dyn -a <HSM IP addresses>
```

OpenSSL Dynamic Engine Provider

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider -a <HSM IP addresses>
```

Key Storage Provider (KSP)

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" -a <HSM IP addresses>
```

JCE provider

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-jce -a <HSM IP addresses>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" -a <HSM IP addresses>
```

CloudHSM CLI

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the HSMs>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI IPv4 / IPv6 addresses of the HSMs>
```

Note

you can use the parameter `--cluster-id` in place of `-a <HSM_IP_ADDRESSES>`. Para ver os requisitos de uso de `--cluster-id`, consulte [AWS CloudHSM Ferramenta de configuração do Client SDK 5](#).

Para obter mais informações sobre o parâmetro `-a`, consulte [the section called “Parâmetros”](#).

Especifique cluster, região e endpoint para o Client SDK 5

Example

Este exemplo usa o parâmetro `cluster-id` para fazer o bootstrap do Client SDK 5 fazendo uma chamada `DescribeClusters`.

PKCS #11 library

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --cluster-id <cluster-1234567>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --cluster-id <cluster-1234567>
```

OpenSSL Dynamic Engine

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --cluster-id <cluster-1234567>
```

Key Storage Provider (KSP)

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --cluster-id <cluster-1234567>
```

JCE provider

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-jce --cluster-id <cluster-1234567>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --cluster-id <cluster-1234567>
```

CloudHSM CLI

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli --cluster-id <cluster-1234567>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5 com **cluster-id**

- Use o ID do cluster `cluster-1234567` para especificar o endereço IP de um HSM no seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --cluster-id <cluster-1234567>
```

Você pode usar os parâmetros `--region` e `--endpoint` em combinação com o parâmetro `cluster-id` para especificar como o sistema faz a chamada `DescribeClusters`. Por exemplo, se a região do cluster for diferente daquela configurada como padrão do AWS CLI, você deve usar o parâmetro `--region` para usar essa região. Além disso, você pode especificar o endpoint da AWS CloudHSM API a ser usado na chamada, o que pode ser necessário para várias configurações de rede, como usar endpoints de interface VPC que não usam o nome de host DNS padrão para AWS CloudHSM

PKCS #11 library

Para fazer o bootstrap de uma instância Linux EC2 com um endpoint e uma região personalizados

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --cluster-id <cluster-1234567> --  
region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Para fazer o bootstrap de uma instância Windows EC2 com um endpoint e uma região

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --cluster-  
id <cluster-1234567>--region <us-east-1> --endpoint <https://cloudhsmv2.us-  
east-1.amazonaws.com>
```

OpenSSL Dynamic Engine

Para fazer o bootstrap de uma instância Linux EC2 com um endpoint e uma região personalizados

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --cluster-id <cluster-1234567> --  
region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Key Storage Provider (KSP)

Para fazer o bootstrap de uma instância Windows EC2 com um endpoint e uma região

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

JCE provider

Para fazer o bootstrap de uma instância Linux EC2 com um endpoint e uma região personalizados

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
$ sudo /opt/cloudhsm/bin/configure-jce --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Para fazer o bootstrap de uma instância Windows EC2 com um endpoint e uma região

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

CloudHSM CLI

Para fazer o bootstrap de uma instância Linux EC2 com um endpoint e uma região personalizados

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
$ sudo /opt/cloudhsm/bin/configure-cli --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Para fazer o bootstrap de uma instância Windows EC2 com um endpoint e uma região

- Use a ferramenta de configuração para especificar o endereço IP de um HSM no seu cluster com uma região e endpoint personalizados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Para obter mais informações sobre os parâmetros `--endpoint`, `--cluster-id` e `--region`, consulte [the section called “Parâmetros”](#).

Atualize o certificado e a chave do cliente para autenticação mútua cliente TLS e HSM

Example

Este exemplo mostra como usar os `--client-key-hsm-tls-file` parâmetros `--client-cert-hsm-tls-file` e para reconfigurar o SSL especificando uma chave personalizada e um certificado SSL para AWS CloudHSM

PKCS #11 library

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
$ sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
    --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
    --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
    --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
    --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

OpenSSL Dynamic Engine

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
    --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
    --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Key Storage Provider (KSP)

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" `
    --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
    --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

JCE provider

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-jce \
    --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
    --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
    --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
```

```
--client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-  
client.key>
```

CloudHSM CLI

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Linux

1. Copie a chave e o certificado para o diretório apropriado.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>  
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-cli \  
--client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \  
--client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Para usar um certificado e uma chave personalizados para autenticação mútua cliente TLS e HSM com o Client SDK 5 no Windows

1. Copie a chave e o certificado para o diretório apropriado.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>  
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Com um PowerShell intérprete, use a ferramenta de configuração para especificar `ssl-client.pem` e `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" \  
--client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-  
client.pem> \  
--client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-  
client.key>
```

Para obter mais informações sobre os parâmetros `--client-cert-hsm-tls-file` e `--client-key-hsm-tls-file`, consulte [the section called “Parâmetros”](#).

Configurações de durabilidade da chave do cliente

Example

Este exemplo usa o parâmetro `--disable-key-availability-check` para desativar as configurações de durabilidade da chave do cliente. Para executar um cluster com um único HSM, você deve desativar as configurações de durabilidade da chave do cliente.

PKCS #11 library

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --disable-key-availability-check
```

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --disable-key-availability-check
```

OpenSSL Dynamic Engine

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --disable-key-availability-check
```

OpenSSL Dynamic Engine Provider

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --disable-key-availability-check
```

Key Storage Provider (KSP)

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --disable-key-availability-check
```

JCE provider

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-jce --disable-key-availability-check
```

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --disable-key-availability-check
```


CloudHSM CLI

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Linux

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
$ sudo /opt/cloudhsm/bin/configure-cli --disable-key-availability-check
```

Para desativar a durabilidade de chave do cliente para o Client SDK 5 no Windows

- Use a ferramenta de configuração para desativar as configurações de durabilidade de chave do cliente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --disable-key-availability-check
```

Para obter mais informações sobre o parâmetro `--disable-key-availability-check`, consulte [the section called “Parâmetros”](#).

Gerenciar opções de registro em logging

Example

O Client SDK 5 usa os parâmetros `log-file`, `log-level`, `log-rotation` e `log-type` para gerenciar o logging.

Note

Para configurar seu SDK para ambientes sem servidor, como AWS Fargate ou AWS Lambda, recomendamos que você configure seu tipo de log como `AWS CloudHSM term`. Os registros do cliente serão enviados `stderr` e capturados no grupo de CloudWatch registros de registros configurado para esse ambiente.

PKCS #11 library

Local dos logs padrão

- Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

Linux

```
/opt/cloudhsm/run/cloudhsm-pkcs11.log
```

Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-pkcs11.log
```

Para configurar o nível de logging e deixar outras opções de logging definidas como padrão

- ```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --log-level info
```

Para configurar as opções de logging de arquivos

- ```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --log-type file --log-file <file name with path> --log-rotation daily --log-level info
```

Para configurar as opções de logging do terminal

- ```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --log-type term --log-level info
```

## OpenSSL Dynamic Engine

### Local dos logs padrão

- Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

#### Linux

```
stderr
```

Para configurar o nível de logging e deixar outras opções de logging definidas como padrão

- ```
$ sudo /opt/cloudhsm/bin/configure-dyn --log-level info
```

Para configurar as opções de logging de arquivos

- ```
$ sudo /opt/cloudhsm/bin/configure-dyn --log-type file --log-file <file name with path> --log-rotation daily --log-level info
```

Para configurar as opções de logging do terminal

- ```
$ sudo /opt/cloudhsm/bin/configure-dyn --log-type term --log-level info
```

OpenSSL Dynamic Engine Provider

Local dos logs padrão

- Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

Linux

```
stderr
```

Para configurar o nível de logging e deixar outras opções de logging definidas como padrão

- ```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --log-level info
```

Para configurar as opções de logging de arquivos

- ```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --log-type file --log-file <file name with path> --log-rotation daily --log-level info
```

Para configurar as opções de logging do terminal

- ```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --log-type term --log-level info
```

## Key Storage Provider (KSP)

Local dos logs padrão

- Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-ksp.log
```

Para configurar o nível de logging e deixar outras opções de logging definidas como padrão

- ```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --log-level info
```

Para configurar as opções de logging de arquivos

- ```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --log-type file --log-file <file name with path> --log-rotation daily --log-level info
```

Para configurar as opções de logging do terminal

- ```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --log-type term --log-level info
```

JCE provider

Local dos logs padrão

- Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

Linux

```
/opt/cloudhsm/run/cloudhsm-jce.log
```

Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-jce.log
```

Para configurar o nível de logging e deixar outras opções de logging definidas como padrão

- ```
$ sudo /opt/cloudhsm/bin/configure-jce --log-level info
```

Para configurar as opções de logging de arquivos

- ```
$ sudo /opt/cloudhsm/bin/configure-jce --log-type file --log-file <file name with path> --log-rotation daily --log-level info
```

Para configurar as opções de logging do terminal

- ```
$ sudo /opt/cloudhsm/bin/configure-jce --log-type term --log-level info
```

## CloudHSM CLI

### Local dos logs padrão

- Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

#### Linux

```
/opt/cloudhsm/run/cloudhsm-cli.log
```

#### Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-cli.log
```

Para configurar o nível de logging e deixar outras opções de logging definidas como padrão

- ```
$ sudo /opt/cloudhsm/bin/configure-cli --log-level info
```

Para configurar as opções de logging de arquivos

- ```
$ sudo /opt/cloudhsm/bin/configure-cli --log-type file --log-file <file name with path> --log-rotation daily --log-level info
```

Para configurar as opções de logging do terminal

- ```
$ sudo /opt/cloudhsm/bin/configure-cli --log-type term --log-level info
```

Para obter mais informações sobre os parâmetros `log-file`, `log-level`, `log-rotation` e `log-type`, consulte [the section called “Parâmetros”](#).

Coloque o certificado de emissão para o Client SDK 5

Example

Este exemplo usa o parâmetro `--hsm-ca-cert` para atualizar a localização do certificado de emissão do Client SDK 5.

PKCS #11 library

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --hsm-ca-cert <customerCA certificate file>
```

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --hsm-ca-cert <customerCA certificate file>
```

OpenSSL Dynamic Engine

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --hsm-ca-cert <customerCA certificate file>
```

OpenSSL Dynamic Engine Provider

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --hsm-ca-cert <customerCA certificate file>
```

Key Storage Provider (KSP)

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --hsm-ca-cert <customerCA certificate file>
```

JCE provider

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-jce --hsm-ca-cert <customerCA certificate file>
```

Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --hsm-ca-cert <customerCA certificate file>
```

CloudHSM CLI

Para colocar o certificado de emissão no Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
$ sudo /opt/cloudhsm/bin/configure-cli --hsm-ca-cert <customerCA certificate file>
```


Para colocar o certificado de emissão no Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar um local para o certificado de emissão.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --hsm-ca-cert <customerCA certificate file>
```

Para obter mais informações sobre o parâmetro `--hsm-ca-cert`, consulte [the section called “Parâmetros”](#).

Provedor OpenSSL Bootstrap

Use a `configure-openssl-provider` ferramenta para inicializar sua instalação do OpenSSL Provider e conectá-la ao seu cluster. AWS CloudHSM

Para inicializar o OpenSSL Provider

1. Execute o `configure-openssl-provider` comando com o endereço IP de um HSM no seu cluster:

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider -a <HSM IP address>
```

<HSM IP address> Substitua pelo endereço IP de qualquer HSM em seu cluster.

2. Verifique a configuração verificando se o provedor OpenSSL pode se conectar ao seu cluster:

```
$ openssl list -providers -provider-path /opt/cloudhsm/lib -provider cloudhsm
```

Para obter mais informações sobre os parâmetros de configuração, consulte [the section called “Parâmetros”](#).

Configurações avançadas para a ferramenta de configuração do Client SDK 5

A ferramenta de configuração AWS CloudHSM do Client SDK 5 inclui configurações avançadas que não fazem parte dos recursos gerais que a maioria dos clientes utiliza. Configurações avançadas fornecem recursos adicionais.

 Important

Depois de fazer qualquer alteração na configuração, você precisará reiniciar o aplicativo para que as alterações entrem em vigor.

- Configurações avançadas para PKCS #11
 - [Configuração de vários slots com a biblioteca PKCS #11 para AWS CloudHSM](#)
 - [Tente novamente os comandos da biblioteca PKCS #11 para AWS CloudHSM](#)
- Configurações avançadas para OpenSSL
 - [Tente novamente os comandos do OpenSSL para AWS CloudHSM](#)
- Configurações avançadas para KSP
 - [SDK3 modo de compatibilidade do Key Storage Provider \(KSP\) para AWS CloudHSM](#)
- Configurações avançadas para JCE
 - [Conectando-se a vários AWS CloudHSM clusters com o provedor JCE](#)
 - [Repita os comandos do JCE para AWS CloudHSM](#)
 - [Extração de chaves usando JCE para AWS CloudHSM](#)
- Configurações avançadas para interface de linha de AWS CloudHSM comando (CLI)
 - [Conectar-se a vários clusters com a CLI do CloudHSM](#)

AWS CloudHSM Tópicos relacionados ao Client SDK 5

Consulte os tópicos relacionados a seguir para saber mais sobre o AWS CloudHSM Client SDK 5.

- [DescribeClusters](#) Operação de API
- [describe-clusters](#) no &CLI;
- [Get-HSM2Cluster](#) PowerShell cmdlet
- [Bootstrap do Client SDK 5](#)
- [Provedor OpenSSL Bootstrap](#)
- [AWS CloudHSM Endpoints da VPC](#)
- [Gerenciando as configurações de durabilidade da chave Client SDK 5](#)
- [Registro em log do Client SDK 5](#)
- [Configurar mTLS \(recomendado\)](#)

AWS CloudHSM Ferramenta de configuração do Client SDK 3

Use a ferramenta de configuração AWS CloudHSM do Client SDK 3 para inicializar o daemon do cliente e configurar o CloudHSM Management Utility (CMU).

Tópicos

- [AWS CloudHSM Sintaxe de configuração do SDK 3 do cliente](#)
- [AWS CloudHSM Parâmetros de configuração do Client SDK 3](#)
- [AWS CloudHSM Exemplos de configuração do Client SDK 3](#)
- [AWS CloudHSM Tópicos relacionados à configuração do Client SDK 3](#)

AWS CloudHSM Sintaxe de configuração do SDK 3 do cliente

A tabela a seguir ilustra a sintaxe dos arquivos de AWS CloudHSM configuração do Client SDK 3.

```
configure -h | --help
-a <ENI IP address>
-m [-i <daemon_id>]
--ssl --pkey <private key file> --cert <certificate file>
--cmu <ENI IP address>
```

AWS CloudHSM Parâmetros de configuração do Client SDK 3

A seguir está uma lista de parâmetros para configurar o AWS CloudHSM Client SDK 3.

-h | --help

Exibe a sintaxe do comando.

Obrigatório: Sim

-uma **<ENI IP address>**

Adiciona o endereço IP da interface de rede elástica (ENI) do HSM aos arquivos de configuração do AWS CloudHSM. Insira o endereço IP ENI de qualquer um dos HSMs no cluster. Não importa qual você seleciona.

Para obter os endereços IP ENI do HSMs em seu cluster, use a [DescribeClusters](#) operação, o AWS CLI comando [describe-clusters](#) ou o cmdlet. [Get-HSM2Cluster](#) PowerShell

Note

Antes de executar o `-a` configure comando, pare o AWS CloudHSM cliente. Em seguida, quando o `-a` comando for concluído, reinicie o AWS CloudHSM cliente. Para obter detalhes, [consulte os exemplos](#).

Esse parâmetro edita os seguintes arquivos de configuração:

- `/opt/cloudhsm/etc/cloudhsm_client.cfg`: usado pelo AWS CloudHSM cliente e pelo [key_mgmt_util](#).
- `/opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg`: usado por [cloudhsm_mgmt_util](#).

Quando o AWS CloudHSM cliente inicia, ele usa o endereço IP ENI em seu arquivo de configuração para consultar o cluster e atualizar o `cluster.info` arquivo (`/opt/cloudhsm/daemon/1/cluster.info`) com os endereços IP ENI corretos para todos HSMs no cluster.

Obrigatório: Sim

`-m`

Atualiza os endereços IP ENI do HSM no arquivo de configuração usado pela CMU.

Note

O parâmetro `-m` é para uso com a CMU do Client SDK 3.2.1 e versões anteriores. Para a CMU do Client SDK 3.3.0 e versões posteriores, consulte o parâmetro `--cmu`, que simplifica o processo de atualização de dados do HSM para CMU.

Quando você atualiza o `-a` parâmetro configure e depois inicia o AWS CloudHSM cliente, o daemon do cliente consulta o cluster e atualiza os `cluster.info` arquivos com os endereços IP HSM corretos para todos HSMs no cluster. A execução do comando `-mconfigure` completa a atualização, copiando os endereços IP do HSM de `cluster.info` para o arquivo de configuração `cloudhsm_mgmt_util.cfg` usado pela `cloudhsm_mgmt_util`.

Certifique-se de executar o `-a` configure comando e reiniciar o AWS CloudHSM cliente antes de executar o `-m` comando. Isso garante que os dados copiados em `cloudhsm_mgmt_util.cfg` de `cluster.info` estejam completos e precisos.

Obrigatório: Sim

-i

Especifica um daemon de cliente alternativo. O valor padrão representa o cliente AWS CloudHSM

Padrão: 1

Exigido: Não

--ssl

Substitui a chave e o certificado SSL para o cluster com a chave privada e o certificado especificados. Ao usar esse parâmetro, os parâmetros --pkey e --cert são necessários.

Obrigatório: não

--pkey

Especifica a nova chave privada. Insira o caminho e o nome do arquivo que contém a chave privada.

Obrigatório: sim, se --ssl for especificado. Caso contrário, isso não deve ser usado.

--cert

Especifica o novo certificado. Insira o caminho e o nome do arquivo que contém o certificado. O certificado deve ser associado ao certificado `customerCA.crt`, o certificado autoassinado usado para inicializar o cluster. Para obter mais informações, consulte [Inicializar o cluster](#).

Obrigatório: sim, se --ssl for especificado. Caso contrário, isso não deve ser usado.

--esperma **<ENI IP address>**

Combina os parâmetros -a e -m em um único parâmetro. Adiciona o endereço IP HSM elastic network interface (ENI) especificado aos arquivos de AWS CloudHSM configuração e, em seguida, atualiza o arquivo de configuração CMU. Insira um endereço IP ENI de um HSM no cluster. Para o Client SDK 3.2.1 e versões anteriores, consulte [Usando a CMU com o Client SDK 3.2.1 e versões anteriores](#).

Obrigatório: Sim

AWS CloudHSM Exemplos de configuração do Client SDK 3

Esses exemplos mostram como usar a configure ferramenta para o AWS CloudHSM Client SDK 3.

Example: atualize os dados do HSM para o AWS CloudHSM cliente e `key_mgmt_util`

Este exemplo usa o `-a` parâmetro de configure para atualizar os dados do HSM para o AWS CloudHSM cliente e `key_mgmt_util`. Para usar o `-a` parâmetro, você deve ter o endereço IP de um deles HSMs em seu cluster. Use o console ou a AWS CLI para obter o endereço IP.

Para obter um endereço IP para um HSM (console)

1. Abra o AWS CloudHSM console em <https://console.aws.amazon.com/cloudhsm/casa>.
2. Para alterar a região da Amazon Web Services, use o seletor de região no canto superior direito da página.
3. Para abrir a página de detalhes do cluster, na tabela do cluster, escolha o ID do cluster.
4. Para obter o endereço IP, vá até a HSMs guia. Para IPv4 clusters, escolha um endereço listado em IPv4 Endereço ENI. Para clusters de pilha dupla, use o endereço ENI IPv4 ou ENI. IPv6

Para obter um endereço IP para um HSM (AWS CLI)

- Obtenha o endereço IP de um HSM usando o comando [describe-clusters](#) do AWS CLI. Na saída do comando, o endereço IP do HSMs são os valores de `EniIp` e `EniIPv6` (se for um cluster de pilha dupla).

```
$ aws cloudhsmv2 describe-clusters
{
  "Clusters": [
    { ... }
    "Hsms": [
      {
...
        "EniIp": "10.0.0.9",
...
      },
      {
...
        "EniIp": "10.0.1.6",
        "EniIPv6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
      }
    ]
  }
}
```

Para atualizar os dados do HSM

1. Antes de atualizar o -a parâmetro, pare o AWS CloudHSM cliente. Isso evita conflitos que podem ocorrer enquanto configure edita o arquivo de configuração do cliente. Se o cliente já estiver parado, esse comando não terá efeitos, então você poderá usá-lo em um script.

Amazon Linux

```
$ sudo stop cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

CentOS 8

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client stop
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client stop
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe stop AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

Use Ctrl + C na janela de comando em que você iniciou o AWS CloudHSM cliente.

2. Esta etapa usa o parâmetro `-a` de `configure` para adicionar o endereço IP da ENI 10.0.0.9 aos arquivos de configurações.

Amazon Linux

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

Amazon Linux 2

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

CentOS 7

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

CentOS 8

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

RHEL 7

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

RHEL 8

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```


Ubuntu 16.04 LTS

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

Ubuntu 18.04 LTS

```
$ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a 10.0.0.9
```

3. Em seguida, reinicie o AWS CloudHSM cliente. Quando o cliente é iniciado, ele usa o endereço IP ENI em seu arquivo de configuração para consultar o cluster. Em seguida, ele grava os endereços IP ENI de todos HSMs no cluster no `cluster.info` arquivo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```

RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe  
C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Quando o comando é concluído, os dados do HSM que o AWS CloudHSM cliente e o `key_mgmt_util` usam estão completos e precisos.

Example: atualize os dados do HSM para CMU a partir do client SDK 3.2.1 e versões anteriores

Esse exemplo usa o comando `-m configure` para copiar os dados de HSM atualizados do arquivo `cluster.info` para o arquivo `cloudhsm_mgmt_util.cfg` usado pela `cloudhsm_mgmt_util`. Use isso com a CMU que vem com o SDK do cliente 3.2.1 e versões anteriores.

- Antes de executar o `-m`, pare o AWS CloudHSM cliente, execute o `-a` comando e reinicie o AWS CloudHSM cliente, conforme mostrado no [exemplo anterior](#). Isso garante que os dados copiados no arquivo `cloudhsm_mgmt_util.cfg` do arquivo `cluster.info` estejam completos e precisos.

Linux

```
$ sudo /opt/cloudhsm/bin/configure -m
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -m
```

Example: atualize os dados do HSM para CMU a partir do Client SDK 3.3.0 e versões posteriores

Este exemplo usa o parâmetro `--cmu` do comando `configure` para atualizar os dados HSM da CMU. Use isso com a CMU que vem com o Client SDK 3.3.0 e versões posteriores. Para obter mais informações sobre o uso da CMU, consulte [Como usar a CloudHSM Management Utility \(CMU\) para gerenciar usuários](#) e [como usar a CMU com o Client SDK 3.2.1 e versões anteriores](#).

- Use o parâmetro `--cmu` para transmitir o endereço IP de um HSM no seu cluster.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

AWS CloudHSM Tópicos relacionados à configuração do Client SDK 3

Consulte os tópicos relacionados a seguir para saber mais sobre o AWS CloudHSM Client SDK 3.

- [Configurar AWS CloudHSM key_mgmt_util](#)

AWS CloudHSM Interface de linha de comando (CLI)

A CLI do CloudHSM ajuda os administradores a gerenciar usuários e os usuários de criptografia a gerenciar chaves no cluster no AWS CloudHSM. A CLI inclui ferramentas que podem ser usadas para criar, excluir e listar usuários, alterar senhas de usuários e atualizar a autenticação multifator

(MFA) de usuários. Ela também inclui comandos que geram, excluem, importam e exportam chaves, obtêm e definem atributos, localizam chaves e executam operações criptográficas.

Para ver uma lista definida de usuários da CLI do CloudHSM, consulte [Gerenciamento de usuários do HSM com a CLI do CloudHSM](#). Para obter uma lista definida dos atributos de chave da CLI do CloudHSM, consulte [Atributos de chave da CloudHSM CLI](#). Para obter informações sobre como usar a CLI do CloudHSM para gerenciar chaves, consulte [Gerenciamento de chaves com a CLI do CloudHSM](#).

Para começar rapidamente, consulte [Introdução à Interface de Linha de AWS CloudHSM Comando \(CLI\)](#). Para obter informações detalhadas sobre comandos da CLI do CloudHSM e exemplos de uso dos comandos, consulte [Referência para comandos da CLI do CloudHSM](#).

Tópicos

- [AWS CloudHSM Plataformas suportadas pela interface de linha de comando \(CLI\)](#)
- [Introdução à Interface de Linha de AWS CloudHSM Comando \(CLI\)](#)
- [Modos de comando na CLI do CloudHSM](#)
- [Atributos de chave da CloudHSM CLI](#)
- [Configurações avançadas da CLI do CloudhSM](#)
- [Referência para comandos da CLI do CloudHSM](#)

AWS CloudHSM Plataformas suportadas pela interface de linha de comando (CLI)

Este tópico descreve as plataformas Linux e Windows suportadas pela AWS CloudHSM CLI.

Suporte a Linux

Plataformas compatíveis	Arquitetura X86_64	Arquitetura ARM
Amazon Linux 2	Sim	Sim
Amazon Linux 2023	Sim	Sim
Red Hat Enterprise Linux 8 (8.3+)	Sim	Sim

Plataformas compatíveis	Arquitetura X86_64	Arquitetura ARM
Red Hat Enterprise Linux 9 (9.2+)	Sim	Sim
Red Hat Enterprise Linux 10 (10.0+)	Sim	Sim
Ubuntu 22.04 LTS	Sim	Sim
Ubuntu 24.04 LTS	Sim	Sim

- O SDK 5.16 foi a última versão a fornecer suporte à plataforma Ubuntu 20.04 LTS. Para obter mais informações, consulte o [site do Ubuntu](#).
- O SDK 5.12 foi a última versão a fornecer suporte à plataforma CentOS 7 (7.8+). Para obter mais informações, consulte o [site do CentOS](#).
- O SDK 5.12 foi a última versão a fornecer suporte à plataforma Red Hat Enterprise Linux 7 (7.8+). Para obter mais informações, acesse o [site do Red Hat](#).
- O SDK 5.4.2 foi a última versão a fornecer suporte à plataforma CentOS 8. Para obter mais informações, consulte o [site do CentOS](#).

Suporte ao Windows

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019
- Microsoft Windows Server 2022
- Microsoft Windows Server 2025

Introdução à Interface de Linha de AWS CloudHSM Comando (CLI)

Com a interface de linha de comando (CLI) da CLI do CloudHSM, você pode gerenciar usuários em seu cluster. AWS CloudHSM Use este tópico para iniciar as tarefas básicas de gerenciamento de usuários do HSM, como criar usuários, listar usuários e conectar a CloudHSM CLI ao cluster.

Tópicos

- [Instalar o cliente do CloudHSM](#)
- [Usar a CloudHSM CLI](#)

Instalar o cliente do CloudHSM

Use os comandos a seguir para baixar e instalar a CLI do CloudHSM para. AWS CloudHSM

Amazon Linux 2023

Amazon Linux 2023 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

Amazon Linux 2023 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Amazon Linux 2 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.x86_64.rpm
```

Amazon Linux 2 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

RHEL 10 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-cli-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.x86_64.rpm
```

RHEL 10 sobre ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-cli-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

RHEL 9 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.x86_64.rpm
```

RHEL 9 sobre ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

RHEL 8 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.x86_64.rpm
```

RHEL 8 sobre ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.aarch64.rpm
```

Ubuntu 24.04 LTS

Ubuntu 24.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-cli_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_amd64.deb
```

Ubuntu 24.04 LTS na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-cli_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Ubuntu 22.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-cli_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_amd64.deb
```

Ubuntu 22.04 LTS na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-cli_latest_u22.04_arm64.deb
```



```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_arm64.deb
```

Windows Server 2022

Para o Windows Server 2022 na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msixexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2019

Para o Windows Server 2019 na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msixexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2016

Para o Windows Server 2016 na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msixexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Use o comando a seguir para configurar a CloudHSM CLI.

Para fazer o bootstrap de uma instância do EC2 do Linux para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the HSMs>
```

Para fazer o bootstrap de uma instância do EC2 do Windows para o Client SDK 5

- Use a ferramenta de configuração para especificar o endereço IP do(s) HSM(s) em seu cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI IPv4 / IPv6 addresses of the HSMs>
```

Usar a CloudHSM CLI

Use os comandos a seguir para iniciar e usar a CloudHSM CLI.

1. Use o comando a seguir para iniciar CloudHSM CLI.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Use o comando login para fazer login no cluster. Todos os usuários podem usar esse comando.

O comando no exemplo a seguir faz login em admin, que é a conta [administradora](#) padrão. A senha desse usuário foi definida quando [ativou o cluster](#).

```
aws-cloudhsm > login --username admin --role admin
```

O sistema solicita que você forneça sua senha. Você insere a senha e a saída mostra que o comando foi bem-sucedido.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin",
    "role": "admin"
  }
}
```

3. Execute o comando `user list` para listar todos os usuários no cluster.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

4. Use `user create` para criar um usuário de CU chamado **example_user**.

Você pode criar CUs porque, em uma etapa anterior, você fez login como usuário administrador. Somente usuários administradores podem realizar tarefas de gerenciamento de usuários, como criar e excluir usuários e alterar as senhas de outros usuários.

```
aws-cloudhsm > user create --username example_user --role crypto-user
```

```
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "example_user",
    "role": "crypto-user"
  }
}
```

5. Use `user list` para listar todos os usuários no cluster.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "example_user",
        "role": "crypto_user",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

6. Use o `logout` comando para sair do AWS CloudHSM cluster.

```
aws-cloudhsm > logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

7. Use o comando quit para parar .

```
aws-cloudhsm > quit
```

Modos de comando na CLI do CloudHSM

Na CLI do CloudHSM, você pode executar comandos de duas maneiras diferentes: no modo de comando único e no modo interativo. O modo interativo foi projetado para usuários e o modo de comando único foi projetado para scripts.

Note

Todos os comandos funcionam no modo interativo e no modo de comando único.

Modo interativo

Use os comandos a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

Ao usar a CLI no modo interativo, você pode fazer login em uma conta de usuário usando o comando do login.

```
aws-cloudhsm > login --username <USERNAME> --role <ROLE>
```

Para listar todos os comandos da CLI do CloudHSM, execute o seguinte comando:

```
aws-cloudhsm > help
```

Para obter a sintaxe de um comando da CLI do CloudHSM, execute o seguinte comando:

```
aws-cloudhsm > help <command-name>
```

Para obter uma lista de usuários no HSMs, digite `user list`.

```
aws-cloudhsm > user list
```

Para encerrar sua sessão da CLI do CloudHSM, execute o seguinte comando:

```
aws-cloudhsm > quit
```

Modo de comando único

Note

Ao usar o modo de comando único, você deve escapar de quaisquer caracteres especiais em variáveis de ambiente e argumentos de linha de comando que possam ser interpretados pelo seu shell.

Se você executar a CLI do CloudHSM usando o modo de comando único, precisará definir duas variáveis de ambiente para fornecer credenciais: `CLOUDHSM_PIN` e `CLOUDHSM_ROLE`:

```
$ export CLOUDHSM_ROLE=admin
```

```
$ export CLOUDHSM_PIN=admin_username:admin_password
```

Depois de fazer isso, você pode executar comandos usando as credenciais armazenadas em seu ambiente.

```
$ cloudhsm-cli user change-password --username alice --role crypto-user  
Enter password:  
Confirm password:  
{
```

```
"error_code": 0,
"data": {
  "username": "alice",
  "role": "crypto-user"
}
```

Atributos de chave da CloudHSM CLI

Este tópico descreve como usar a CloudHSM CLI para definir atributos de chave. Um atributo de chave na CloudHSM CLI pode definir o tipo de uma chave, como ela pode funcionar ou como será rotulada. Alguns atributos definem características exclusivas (o tipo de uma chave, por exemplo). Outros atributos podem ser definidos como verdadeiros ou falsos. Alterá-los ativa ou desativa uma parte da funcionalidade da chave.

Para ver exemplos de como usar os atributos de chave, consulte os comandos listados no comando principal [A key na CLI do CloudHSM](#).

Os tópicos a seguir fornecem mais detalhes sobre os principais atributos na CloudHSM CLI.

Tópicos

- [Atributos compatíveis da CloudHSM CLI](#)
- [Verificar o valor na CloudHSM CLI](#)
- [Tópicos relacionados à CloudHSM CLI](#)

Atributos compatíveis da CloudHSM CLI

Como melhor prática, defina valores somente para os atributos que deseja tornar restritivos. Se você não especificar um valor, o CloudHSM utilizará o valor padrão especificado na tabela abaixo.

A tabela a seguir lista os atributos de chave, valores possíveis, padrões e notas relacionadas da CloudHSM CLI. Uma célula vazia na coluna Valor indica que não há nenhum valor padrão específico atribuído ao atributo.

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
always-sensitive	O valor é True se sensitive	Não	Não

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
	sempre esteve definido como True e nunca foi alterado.		
check-value	Valor de verificação da chave. Para obter mais informações, consulte Detalhes adicionais .	Não	Não
class	Valores possíveis : secret-key , public-key e private-key .	Não	Sim
curve	Curva elíptica usada para gerar o par de chaves EC. Valores válidos: secp224r1 secp256r1 ,prime256v1 ,secp384r1 ,secp256k1 ,secp521r1 , e ed25519 ed25519só é compatível com instâncias hsm2m.medium no modo não FIPS.	Não	Configurável com EC, não configurável com RSA
decrypt	Padrão: False	Sim	Sim

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
derive	Padrão: False	A derivação pode ser definida nas instâncias hsm2m.medium. Ela não pode ser definida para chaves RSA em instâncias hsm1.medium.	Sim
destroyable	Padrão: True	Sim	Sim
ec-point	Para chaves EC, codificação DER do ECPoint valor ANSI X9.62 "Q" em formato hexadecimal. Para outros tipos de chave, esse atributo não existe.	Não	Não
encrypt	Padrão: False	Sim	Sim
extractable	Padrão: True	Não	Sim
id	Padrão: Vazio	id pode ser definido em todas as instâncias hsm2m.medium. Ela não pode ser definida em instâncias hsm1.medium.	Sim
key-length-bytes	Necessário para gerar uma chave AES. Valores válidos: 16, 24 e 32 bytes.	Não	Não

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
key-type	Valores possíveis : aes, rsa e ec.	Não	Sim
label	Padrão: Vazio	Sim	Sim
local	Padrão: True para chaves geradas no HSM, False para chaves importadas para o HSM.	Não	Não
modifiable	Padrão: True	Pode ser alterado de verdadeiro para falso, mas não de falso para verdadeiro.	Sim
modulus	O módulo que foi usado para gerar um par de chaves RSA. Para outros tipos de chave, esse atributo não existe.	Não	Não
modulus-size-bits	Necessário para gerar um par de chaves RSA. Valor mínimo de 2048.	Não	Configurável com RSA, não configurável com EC

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
<code>never-extractable</code>	O valor é True se extraível nunca tiver sido definido como False. O valor é False se extraível já tiver sido definido como True.	Não	Não
<code>private</code>	Padrão: True	Não	Sim
<code>public-exponent</code>	Necessário para gerar um par de chaves RSA. Valores válidos: o valor deve ser um número ímpar maior que ou igual a 65537.	Não	Configurável com RSA, não configurável com EC
<code>sensitive</code>	Padrão: - O valor é True para chaves AES e chaves privadas EC e RSA. - O valor é False para chaves públicas EC e RSA.	Não	Configurável com chaves privadas, não configurável com chaves públicas.

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
sign	Padrão: - O valor é True para chaves AES. - O valor é False para chaves RSA e EC.	Sim	Sim
token	Padrão: True	Pode ser alterado de falso para verdadeiro, mas não de verdadeiro para falso.	Sim
trusted	Padrão: False	Somente usuários administradores podem definir esse parâmetro.	Não
unwrap	Padrão: False	Sim	Sim, exceto para chaves públicas.
unwrap-template	Os valores devem usar o modelo de atributo aplicado a todas as chaves desencapsuladas com essa chave de encapsulamento.	Sim	Não

Atributo da CloudHSM CLI	Valor	Modificável com a key set-attribute	Configurável na criação da chave
verify	Padrão: - O valor é True para chaves AES. - O valor é False para chaves RSA e EC.	Sim	Sim
wrap	Padrão: False	Sim	Sim, exceto para chaves privadas.
wrap-template	Os valores devem usar o modelo de atributo para corresponder à chave agrupada usando essa chave de agrupamento.	Sim	Não
wrap-with-trusted	Padrão: False	Sim	Sim

Verificar o valor na CloudHSM CLI

O valor de verificação na CloudHSM CLI é um hash de 3 bytes ou uma soma de verificação de uma chave gerada quando o HSM importa ou gera uma chave. Você também pode calcular um valor de verificação fora do HSM, como depois de exportar uma chave. Em seguida, você pode comparar os valores do valor de verificação para confirmar a identidade e a integridade da chave. Para obter o valor de verificação de uma chave, use a [lista de chaves](#) com o sinalizador detalhado.

AWS CloudHSM usa os seguintes métodos padrão para gerar um valor de verificação:

- Chaves simétricas: primeiros 3 bytes do resultado da criptografia de um bloco zero com a chave.
- Pares de chaves assimétricas: primeiros 3 bytes do hash SHA-1 da chave pública.

- Chaves HMAC: o KCV para chaves HMAC não é suportado no momento.

Tópicos relacionados à CloudHSM CLI

Consulte os tópicos a seguir para obter mais informações sobre a CloudHSM CLI.

- [A key na CLI do CloudHSM](#)
- [Referência para comandos da CLI do CloudHSM](#)

Configurações avançadas da CLI do CloudhSM

A interface de linha de AWS CloudHSM comando (CLI) inclui a seguinte configuração avançada, que não faz parte das configurações gerais que a maioria dos clientes utiliza. Essas configurações fornecem recursos adicionais.

- [Conectar-se a vários clusters](#)

Conectar-se a vários clusters com a CLI do CloudHSM

Com o AWS CloudHSM Client SDK 5, você pode configurar a CLI do CloudHSM para permitir conexões com vários clusters do CloudHSM a partir de uma única instância da CLI.

Os tópicos a seguir descrevem como usar a funcionalidade de vários clusters da CLI do CloudHSM para se conectar a vários clusters.

Tópicos

- [Pré-requisitos de vários clusters para AWS CloudHSM](#)
- [Configurar a CLI do CloudHSM para funcionalidade de vários clusters](#)
- [Adicione um cluster à sua AWS CloudHSM configuração](#)
- [Remover um cluster da sua AWS CloudHSM configuração](#)
- [Interaja com vários clusters em AWS CloudHSM](#)

Pré-requisitos de vários clusters para AWS CloudHSM

Antes de configurar seu cluster AWS CloudHSM para se conectar a vários clusters, você deve atender aos seguintes pré-requisitos:

- Dois ou mais AWS CloudHSM clusters aos quais você gostaria de se conectar, junto com seus certificados de cluster.
- Uma EC2 instância com grupos de segurança configurados corretamente para se conectar a todos os clusters acima. Para obter mais informações sobre como configurar um cluster e a instância cliente, consulte [Introdução ao AWS CloudHSM](#).
- Para configurar a funcionalidade de vários clusters, você já deve ter baixado e instalado a CLI do CloudHSM. Se ainda não tiver feito isso, consulte as instruções em [???](#).
- Não será possível acessar um cluster configurado com `./configure-cli [.exe] -a`, pois ele não estará associado a um `cluster-id`. Para reconfigurá-lo, siga as instruções contidas em `config-cli add-cluster`, conforme descrito neste guia.

Configurar a CLI do CloudHSM para funcionalidade de vários clusters

Para configurar a CLI do CloudHSM para a funcionalidade de vários clusters, siga estas etapas:

1. Identifique os clusters aos quais você quer se conectar.
2. Adicione esses clusters à configuração da CLI do CloudHSM usando o subcomando `add-cluster` de [configure-cli](#), conforme descrito abaixo.
3. Reinicie todos os processos da CLI do CloudHSM para que a nova configuração entre em vigor.

Adicione um cluster à sua AWS CloudHSM configuração

Ao se conectar a vários clusters, use o comando `configure-cli add-cluster` para adicionar um cluster à sua configuração.

Sintaxe

```
configure-cli add-cluster [OPTIONS]
  --cluster-id <CLUSTER ID>
  [--region <REGION>]
  [--endpoint <ENDPOINT>]
  [--hsm-ca-cert <HSM CA CERTIFICATE FILE>]
  [--client-cert-hsm-tls-file <CLIENT CERTIFICATE FILE>]
  [--client-key-hsm-tls-file <CLIENT KEY FILE>]
  [-h, --help]
```

Exemplos

Adicione um cluster usando o parâmetro **cluster-id**

Example

Use o parâmetro `configure-cli add-cluster` junto com `cluster-id` para adicionar um cluster (com o ID `cluster-1234567`) para a sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli add-cluster --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" add-cluster --cluster-id <cluster-1234567>
```

Tip

Se o uso de `configure-cli add-cluster` com o parâmetro `cluster-id` não resultar na adição do cluster, consulte o exemplo a seguir para obter uma versão mais longa deste comando que também requer os parâmetros `--region` e `--endpoint` para identificar o cluster que está sendo adicionado. Se, por exemplo, a região do cluster for diferente daquela configurada como padrão da AWS CLI, você deverá usar o parâmetro `--region` para usar a região correta. Além disso, você pode especificar o endpoint da AWS CloudHSM API a ser usado na chamada, o que pode ser necessário para várias configurações de rede, como usar endpoints de interface VPC que não usam o nome de host DNS padrão para. AWS CloudHSM

Adicione um cluster usando os parâmetros **cluster-id**, **endpoint** e **region**

Example

Use `configure-cli add-cluster` junto com os parâmetros `cluster-id`, `endpoint` e `region` para adicionar um cluster (com o ID de `cluster-1234567`) à sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli add-cluster --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" add-cluster --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Para obter mais informações sobre os parâmetros `--endpoint`, `--cluster-id` e `--region`, consulte [the section called “Parâmetros”](#).

Parâmetros

`--ID do cluster` **<Cluster ID>**

Faz uma chamada `DescribeClusters` para encontrar todos os endereços IP da interface de rede elástica (ENI) do HSM no cluster associado ao ID do cluster. O sistema adiciona os endereços IP ENI aos arquivos de AWS CloudHSM configuração.

Note

Se você usar o `--cluster-id` parâmetro de uma EC2 instância dentro de uma VPC que não tem acesso à Internet pública, deverá criar uma interface VPC endpoint à qual se conectar. AWS CloudHSM Para obter mais informações sobre os endpoints da VPC, consulte [???](#).

Obrigatório: Sim

`--ponto final` **<Endpoint>**

Especifique o endpoint AWS CloudHSM da API usado para fazer a `DescribeClusters` chamada. Você deve definir essa opção em combinação com `--cluster-id`.

Obrigatório: não

`--hsm-ca-cert` **<HsmCA Certificate Filepath>**

Especifica o caminho de arquivo do certificado de CA do HSM.

Obrigatório: não

--região **<Region>**

Especifique a região do seu cluster. Você deve definir essa opção em combinação com --cluster-id.

Se você não fornecer o parâmetro --region, o sistema escolherá a região tentando ler as variáveis de ambiente AWS_DEFAULT_REGION ou AWS_REGION. Se essas variáveis não estiverem definidas, o sistema verificará a região associada ao seu perfil no seu arquivo de AWS Config (normalmente ~/.aws/config), a menos que você tenha especificado um arquivo diferente na variável de ambiente AWS_CONFIG_FILE. Se nenhuma das opções acima for definida, o sistema usará a região us-east-1 como padrão.

Obrigatório: não

-- client-cert-hsm-tls -arquivo **<client certificate hsm tls path>**

Caminho para o certificado de cliente usado para autenticação mútua entre cliente TLS e HSM.

Use essa opção somente se você tiver registrado pelo menos uma âncora de confiança no HSM com a CLI do CloudHSM. Você deve definir essa opção em combinação com --client-key-hsm-tls-file.

Obrigatório: não

-- client-key-hsm-tls -arquivo **<client key hsm tls path>**

Caminho para a chave do cliente usado para autenticação mútua entre cliente TLS e HSM.

Use essa opção somente se você tiver registrado pelo menos uma âncora de confiança no HSM com a CLI do CloudHSM. Você deve definir essa opção em combinação com --client-cert-hsm-tls-file.

Obrigatório: não

Remover um cluster da sua AWS CloudHSM configuração

Ao se conectar a vários clusters com a CLI do CloudHSM, use o comando `configure-cli remove-cluster` para remover um cluster da sua configuração.

Sintaxe

```
configure-cli remove-cluster [OPTIONS]
```

```
--cluster-id <CLUSTER ID>  
[-h, --help]
```

Exemplos

Remover um cluster usando o parâmetro **cluster-id**

Example

Use o parâmetro `configure-cli remove-cluster` junto com `cluster-id` para remover um cluster (com o ID de `cluster-1234567`) da sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli remove-cluster --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" remove-cluster --  
cluster-id <cluster-1234567>
```

Para obter mais informações sobre o parâmetro `--cluster-id`, consulte [the section called “Parâmetros”](#).

Parâmetro

--ID do cluster <**Cluster ID**>

O ID do cluster a ser removido da configuração.

Obrigatório: Sim

Interaja com vários clusters em AWS CloudHSM

Depois de configurar vários clusters com a CLI do CloudHSM, use o comando `cloudhsm-cli` para interagir com eles.

Exemplos

Definição de um **cluster-id** padrão ao usar o modo interativo

Example

Use o parâmetro [???](#) junto com `cluster-id` para definir um cluster padrão (com o ID de `cluster-1234567`) da sua configuração.

Linux

```
$ cloudhsm-cli interactive --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive --cluster-id <cluster-1234567>
```

Configuração do **cluster-id** ao executar um único comando

Example

Use o parâmetro `cluster-id` para definir o cluster (com o ID de `cluster-1234567`) do qual [???](#) será obtido.

Linux

```
$ cloudhsm-cli cluster hsm-info --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" cluster hsm-info --cluster-id <cluster-1234567>
```

Referência para comandos da CLI do CloudHSM

A CLI do CloudHSM ajuda os administradores a gerenciar usuários em seu cluster. AWS CloudHSM A CLI do CloudHSM pode ser executada em dois modos: modo interativo e modo de comando único. Para começar rapidamente, consulte [Introdução à Interface de Linha de AWS CloudHSM Comando \(CLI\)](#).

Para executar a maioria dos comandos da CLI do CloudHSM, você deve iniciar a CLI do CloudHSM e fazer login no HSM. Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CloudHSM CLI. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Os seguintes tópicos descrevem comandos na CLI do CloudHSM:

Command	Description	Tipo de usuário
activate	Ativa um cluster CloudHSM e confirma que o cluster é novo. Isso deve ser feito antes que qualquer outra operação possa ser executada.	Administrador desativado
hsm-info	Liste os HSMs em seu cluster.	Tudo 1 , incluindo usuários não autenticados. O login não é necessário.
ecdsa	Gera uma assinatura usando uma chave privada EC e o mecanismo de assinatura ECDSA.	Usuários de criptografia (CU)
ed25519ph	Gera uma assinatura usando uma chave privada Ed25519 e o mecanismo de assinatura HashEd DSA.	CU
rsa-pkcs	Gera uma assinatura usando uma chave privada RSA e o mecanismo de assinatura RSA-PKCS.	CU
rsa-pkcs-pss	Gera uma assinatura usando uma chave privada RSA e o mecanismo de RSA-PKCS-PSS assinatura.	CU

Command	Description	Tipo de usuário
ecdsa	Confirma que um arquivo foi assinado no HSM por uma determinada chave pública. Verifica se a assinatura foi gerada usando o mecanismo de assinatura ECDSA. Compara um arquivo assinado com um arquivo de origem e determina se os dois estão criptograficamente relacionados com base em uma determinada chave pública ecdsa e em um mecanismo de assinatura.	CU
ed25519ph	Verifica as assinaturas HashEd do DSA usando uma chave pública Ed25519.	CU
rsa-pkcs	Confirma que um arquivo foi assinado no HSM por uma determinada chave pública. Verifica se a assinatura foi gerada usando o mecanismo de assinatura RSA-PKCS. Compara um arquivo assinado com um arquivo de origem e determina se os dois estão criptograficamente relacionados com base em uma determinada chave pública RSA e em um mecanismo de assinatura.	CU

Command	Description	Tipo de usuário
<u>rsa-pkcs-pss</u>	Confirma que um arquivo foi assinado no HSM por uma determinada chave pública. Verifica se a assinatura foi gerada usando o mecanismo de RSA-PKCS-PSS assinatura a. Compara um arquivo assinado com um arquivo de origem e determina se os dois estão criptograficamente relacionados com base em uma determinada chave pública RSA e em um mecanismo de assinatura.	CU
<u>excluir chave</u>	Exclui uma chave do seu AWS CloudHSM cluster.	CU
<u>key generate-file</u>	Gera um arquivo de chave no seu AWS CloudHSM cluster.	CU
<u>chave generate-asymmetric-pair rsa</u>	Gera um par de chaves RSA assimétrico em seu cluster. AWS CloudHSM	CU
<u>chave, generate-asymmetric-pair etc.</u>	Gera um par de chaves assimétrica de curva elíptica (EC) em seu cluster. AWS CloudHSM	CU
<u>key generate-symmetric aes</u>	Gera uma chave AES simétrica em seu AWS CloudHSM cluster.	CU

Command	Description	Tipo de usuário
key generate-symmetric generic-secret	Gera uma chave secreta genérica simétrica em seu AWS CloudHSM cluster.	CU
pem de importação de chaves	Importa uma chave no formato PEM para um HSM. Você pode usá-lo para importar chaves públicas que foram geradas fora do HSM.	CU
lista de chaves	Encontra todas as chaves do usuário atual presente no seu AWS CloudHSM cluster.	CU
replicação de chave	Replique uma chave de um cluster de origem para um cluster de destino clonado.	CU
atributo do conjunto de chaves	Define os atributos das chaves no seu AWS CloudHSM cluster.	CUs podem executar esse comando, os administradores podem definir o atributo confiável.
compartilhamento de chaves	Compartilha uma chave com outras CUs pessoas em seu AWS CloudHSM cluster.	CU
descompartilhar chave	Cancela o compartilhamento de uma chave com outra CUs em seu AWS CloudHSM cluster.	CU

Command	Description	Tipo de usuário
aes-gcm	Desencapsula uma chave de carga útil no cluster usando a chave de encapsulamento AES e o mecanismo de desencapsulamento AES-GCM.	CU
aes-no-pad	Desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o mecanismo de desempacotamento. AES-NO-PAD	CU
aes-pkcs5-pad	Desempacota uma chave de carga útil usando a chave de empacotamento AES e o mecanismo de desempacotamento AES- PKCS5 -PAD.	CU
aes-zero-pad	Desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o mecanismo de desempacotamento. AES-ZERO-PAD	CU
cloudhsm-aes-gcm	Desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o mecanismo de desempacotamento. CLOUDHSM-AES-GCM	CU

Command	Description	Tipo de usuário
rsa-aes	Desencapsula uma chave de carga útil usando uma chave privada RSA e o mecanismo de desencapsulamento RSA-AES.	CU
rsa-oaep	Desencapsula uma chave de carga útil usando a chave privada RSA e o mecanismo de desencapsulamento RSA-OAEP.	CU
rsa-pkcs	Desencapsula uma chave de carga útil usando a chave privada RSA e o mecanismo de desencapsulamento RSA-PKCS.	CU
aes-gcm	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de encapsulamento AES-GCM.	CU
aes-no-pad	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de empacotamento AES-NO-PAD.	CU
aes-pkcs5-pad	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de empacotamento AES-PKCS5 -PAD.	CU

Command	Description	Tipo de usuário
aes-zero-pad	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de empacotamento AES-ZERO-PAD.	CU
cloudhsm-aes-gcm	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de empacotamento CLOUDHSM-AES-GCM.	CUs
rsa-aes	Encapsula uma chave de carga útil usando uma chave pública RSA no HSM e o mecanismo de encapsulamento RSA-AES.	CU
rsa-oaep	Encapsula uma chave de carga útil usando uma chave pública RSA no HSM e o mecanismo de encapsulamento RSA-OAEP.	CU

Command	Description	Tipo de usuário
Use o comando <code>key wrap rsa-pkcs</code> na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave pública RSA no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento RSA-PKCS. O atributo <code>extractable</code> da chave de carga útil deve ser definido como <code>true</code>. Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas. Para usar o <code>key wrap rsa-pkcs</code> comando, primeiro você deve ter uma chave RSA em seu AWS CloudHSM cluster. Você pode gerar um par de chaves RSA usando o comando <code>generate-asymmetric-pair</code> categoria na CLI do CloudHSM e o atributo <code>wrap</code> definido como <code>true</code>. Tipo de usuário Os seguintes tipos de usuários podem executar este comando. • Usuários de criptomoedas (CUs) Requisitos	Encapsula uma chave de carga útil usando uma chave pública RSA no HSM e o mecanismo de encapsulamento RSA-PKCS.	CU

Command	Description	Tipo de usuário
login	Faça login no seu AWS CloudHSM cluster.	Administrador, usuário de criptografia (CU) e usuário do dispositivo (AU)
logout	Saia do seu AWS CloudHSM cluster.	Administrador, CU e usuário do dispositivo (AU)
quorum token-sign delete	Exclui um ou mais tokens de um serviço autorizado por quórum.	Administrador
quorum token-sign generate	Gera um token para um serviço autorizado por quórum.	Administrador
quorum token-sign list	Lista todos os tokens de quórum de assinatura de token presentes no seu cluster do CloudHSM.	Tudo 1 , incluindo usuários não autenticados. O login não é necessário.
sinal simbólico de quórum list-quorum-values	Lista os valores de quorum definidos no seu cluster CloudHSM.	Tudo 1 , incluindo usuários não autenticados. O login não é necessário.
sinal simbólico de quórum set-quorum-value	Define um novo valor de quórum para um serviço autorizado por quórum.	Administrador
user change-mfa	Altera a estratégia de autenticação multifator (MFA) do usuário.	Administrador, CU

Command	Description	Tipo de usuário
alterar senha de usuário	Altera as senhas dos usuários no HSMs. Qualquer usuário pode alterar sua própria senha. Os administradores podem mudar a senha de qualquer pessoa.	Administrador, CU
criar usuário	Cria um usuário no seu AWS CloudHSM cluster.	Administrador
excluir usuário	Exclui um usuário no seu AWS CloudHSM cluster.	Administrador
lista de usuários	Lista os usuários em seu AWS CloudHSM cluster.	Tudo ¹ , incluindo usuários não autenticados. O login não é necessário.
user change-quorum token-sign register	Registra a estratégia de quórum de assinatura de token de quórum para um usuário.	Administrador

Anotações

- [1] Todos os usuários incluem todas as funções listadas e usuários não logados.

A categoria cluster na CLI do CloudHSM

Na CLI do CloudHSM, cluster é uma categoria principal para um grupo de comandos que, quando combinado com a categoria principal, cria um comando específico para clusters. Atualmente, a categoria cluster consiste nos seguintes comandos:

Tópicos

- [Ativar um cluster com a CLI do CloudHSM](#)
- [Lista HSMs com a CLI do CloudHSM](#)

- [A categoria cluster mtls na CLI do CloudHSM](#)

Ativar um cluster com a CLI do CloudHSM

Use o comando `cluster activate` na CLI do CloudHSM para [ativar um novo cluster](#) no AWS CloudHSM. Esse comando deve ser executado para que o cluster possa ser usado para executar operações criptográficas.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Administrador desativado

Sintaxe

Este comando não possui parâmetros.

```
aws-cloudhsm > help cluster activate
```

```
Activate a cluster
```

```
This command will set the initial Admin password. This process will cause your CloudHSM cluster to move into the ACTIVE state.
```

USAGE:

```
cloudhsm-cli cluster activate [OPTIONS] [--password <PASSWORD>]
```

Options:

```
--cluster-id <CLUSTER_ID>
```

```
Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error
```

```
--password <PASSWORD>
```

```
Optional: Plaintext activation password If you do not include this argument you will be prompted for it
```

```
-h, --help
```

```
Print help (see a summary with '-h')
```

Exemplo

Esse comando ativa seu cluster definindo a senha inicial para seu usuário administrador.

```
aws-cloudhsm > cluster activate
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": "Cluster activation successful"
}
```

Tópicos relacionados

- [criar usuário](#)
- [excluir usuário](#)
- [alterar senha de usuário](#)

Lista HSMs com a CLI do CloudHSM

Use o `cluster hsm-info` comando na CLI do CloudHSM para listar os HSMs módulos de segurança de hardware () em seu cluster. AWS CloudHSM Você não precisa estar conectado na CLI do CloudHSM para executar esse comando.

Note

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração que o AWS CloudHSM cliente e as ferramentas de linha de comando usam. Caso contrário, as alterações feitas podem não ser efetivas HSMs em todas as partes do cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

```
aws-cloudhsm > help cluster hsm-info
```


List info about each HSM in the cluster

Usage: cloudhsm-cli cluster hsm-info [OPTIONS]

Options:

--cluster-id *<CLUSTER_ID>* Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

-h, --help Print help

Exemplo

Esse comando lista o HSMs presente em seu AWS CloudHSM cluster.

```
aws-cloudhsm > cluster hsm-info
{
  "error_code": 0,
  "data": {
    "hsm": [
      {
        "vendor": "Marvell Semiconductors, Inc.",
        "model": "NITROX-III CNN35XX-NFBE",
        "serial-number": "5.3G1941-ICM000590",
        "hardware-version-major": "5",
        "hardware-version-minor": "3",
        "firmware-version-major": "2",
        "firmware-version-minor": "6",
        "firmware-build-number": "16",
        "firmware-id": "CNN35XX-NFBE-FW-2.06-16"
        "fips-state": "2 [FIPS mode with single factor authentication]"
      },
      {
        "vendor": "Marvell Semiconductors, Inc.",
        "model": "NITROX-III CNN35XX-NFBE",
        "serial-number": "5.3G1941-ICM000625",
        "hardware-version-major": "5",
        "hardware-version-minor": "3",
        "firmware-version-major": "2",
        "firmware-version-minor": "6",
        "firmware-build-number": "16",
        "firmware-id": "CNN35XX-NFBE-FW-2.06-16"
        "fips-state": "2 [FIPS mode with single factor authentication]"
      }
    ]
  }
}
```

```
    },  
    {  
      "vendor": "Marvell Semiconductors, Inc.",  
      "model": "NITROX-III CNN35XX-NFBE",  
      "serial-number": "5.3G1941-ICM000663",  
      "hardware-version-major": "5",  
      "hardware-version-minor": "3",  
      "firmware-version-major": "2",  
      "firmware-version-minor": "6",  
      "firmware-build-number": "16",  
      "firmware-id": "CNN35XX-NFBE-FW-2.06-16"  
      "fips-state": "2 [FIPS mode with single factor authentication]"  
    }  
  ]  
}  
}
```

O objeto tem os seguintes atributos:

- **Fornecedor:** o nome do fornecedor do HSM.
- **Modelo:** o número do modelo do HSM.
- **Número de série:** o número de série do dispositivo. Isso pode mudar devido a substituições.
- **Hardware-version-major:** A versão principal do hardware.
- **Hardware-version-minor:** A versão secundária do hardware.
- **Firmware-version-major:** A versão principal do firmware.
- **Firmware-version-minor:** A versão secundária do firmware.
- **Firmware-build-number:** O número de compilação do firmware.
- **Firmware-id:** o ID do firmware, que inclui as versões principais e secundárias junto com a compilação.
- **Estado FIPS:** o modo FIPS do cluster e o que está nele. HSMs Se estiver no modo FIPS, a saída será “2 [FIPS mode with single factor authentication]”. Se estiver no modo não FIPS, a saída será “0 [non-FIPS mode with single factor authentication]”.

Tópicos relacionados

- [Ativar um cluster com a CLI do CloudHSM](#)

A categoria cluster mtls na CLI do CloudHSM

Na cluster mtls CLI do CloudHSM, é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando específico para clusters. AWS CloudHSM Atualmente, essa categoria consiste nos seguintes comandos:

Tópicos

- [Cancelar o registro de uma âncora de confiança com a CLI do CloudHSM](#)
- [Obter o nível de imposição do mTLS com a CloudHSM CLI](#)
- [Listar âncoras de confiança com o CloudHSM CLI](#)
- [Registrar uma âncora de confiança com a CLI do CloudHSM](#)
- [Definir o nível de imposição do mTLS com a CloudHSM CLI](#)

Cancelar o registro de uma âncora de confiança com a CLI do CloudHSM

Use o comando `cluster mtls deregister-trust-anchor` na CLI do CloudHSM para cancelar o registro de uma âncora de confiança para TLS mútuo entre o cliente e o AWS CloudHSM.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador

Requisitos

- Para executar esse comando, é necessário se conectar como usuário administrador.

Sintaxe

```
aws-cloudhsm > help cluster mtls deregister-trust-anchor
```

```
Deregister a trust anchor for mtls
```

```
Usage: cluster mtls deregister-trust-anchor [OPTIONS] --certificate-reference  
[<CERTIFICATE_REFERENCE>...]
```

```
Options:
```

```

--certificate-reference <CERTIFICATE_REFERENCE> A hexadecimal or decimal
certificate reference
--cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the
config file to run the operation against. If not provided, will fall back to the value
provided when interactive mode was started, or error
--approval <APPROVAL> Filepath of signed quorum token file to approve operation
-h, --help Print help

```

Exemplo

Example

No exemplo a seguir, esse comando remove uma âncora de confiança do HSM.

```

aws-cloudhsm > cluster mtls deregister-trust-anchor --certificate-reference 0x01

{
  "error_code": 0,
  "data": {
    "message": "Trust anchor with reference 0x01 deregistered successfully"
  }
}

```

Em seguida, você pode executar o comando `list-trust-anchors` para confirmar que o registro da âncora de confiança foi cancelado do AWS CloudHSM:

```

aws-cloudhsm > cluster mtls list-trust-anchors

{
  "error_code": 0,
  "data": {
    "trust_anchors": []
  }
}

```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<CERTIFICATE_REFERENCE>

Uma referência de certificado hexadecimal ou decimal.

Obrigatório: sim

Warning

Depois de cancelar o registro de uma âncora de confiança no cluster, todas as conexões mTLS existentes usando o certificado do cliente assinado por essa âncora de confiança serão descartadas.

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de cluster for maior que 1.

Tópicos relacionados

- [cluster mtls reregister-trust-anchor](#)
- [cluster mtls list-trust-anchors](#)
- [Configurar mTLS \(recomendado\)](#)

Obter o nível de imposição do mTLS com a CloudHSM CLI

Use o comando `cluster mtls get-enforcement` na CloudHSM CLI para obter o nível de imposição do uso do TLS mútuo entre o cliente e o AWS CloudHSM.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar logado como usuário administrador ou usuário criptográfico (CUs).

Sintaxe

```
aws-cloudhsm > help cluster mtls get-enforcement
```

Get the status of mtls enforcement in the cluster

Usage: cluster mtls get-enforcement [OPTIONS]

Options:

`--cluster-id <CLUSTER_ID>` Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`-h, --help` Print help

Exemplo

Example

No exemplo a seguir, esse comando lista o nível de imposição do mtls do AWS CloudHSM.

```
aws-cloudhsm > cluster mtls get-enforcement
```

```
{
  "error_code": 0,
  "data": {
    "mtls-enforcement-level": "none"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

Tópicos relacionados

- [cluster mtls set-enforcement](#)
- [Configurar mTLS \(recomendado\)](#)

Listar âncoras de confiança com o CloudHSM CLI

Use o comando `cluster mtls list-trust-anchors` na CloudHSM CLI para listar todas as âncoras de confiança que podem ser usadas para TLS mútuo entre o cliente e o AWS CloudHSM.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

```
aws-cloudhsm > help cluster mtls list-trust-anchors
```

```
List all trust anchors for mtls
```

```
Usage: cluster mtls list-trust-anchors [OPTIONS]
```

```
Options:
```

```
--cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the  
config file to run the operation against. If not provided, will fall back to the value  
provided when interactive mode was started, or error
```

```
-h, --help                Print help
```

Exemplo

Example

No exemplo a seguir, esse comando lista todas as âncoras de confiança registradas do AWS CloudHSM.

```
aws-cloudhsm > cluster mtls list-trust-anchors
```

```
{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
        "certificate": "<PEM Encoded Certificate 1>",
        "cluster-coverage": "full"
      },
      {
        "certificate-reference": "0x02",
        "certificate": "<PEM Encoded Certificate 2>",
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

Tópicos relacionados

- [cluster mtls reregister-trust-anchor](#)
- [cluster mtls deregister-trust-anchor](#)
- [Configurar mTLS \(recomendado\)](#)

Registrar uma âncora de confiança com a CLI do CloudHSM

Use o comando `cluster mtls register-trust-anchor` na CLI do CloudHSM para registrar uma âncora de confiança para TLS mútuo entre o cliente e o AWS CloudHSM.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador

Requisitos

AWS CloudHSM Aceita âncoras de confiança com os seguintes tipos de chave:

Tipo de chave	Description
EC	Curvas secp256r1 (P-256), secp384r1 (P-384) e secp521r1 (P-521).
RSA	Chaves RSA de 2.048 bits, 3.072 bits e 4.096 bits.

Sintaxe

```
aws-cloudhsm > help cluster mtls register-trust-anchor
```

Register a trust anchor for mtls

Usage: cluster mtls register-trust-anchor [OPTIONS] --path [**<PATH>**...]

Options:

--cluster-id **<CLUSTER_ID>** Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--path **<PATH>** Filepath of the trust anchor to register

--approval **<APPROVAL>** Filepath of signed quorum token file to approve operation

-h, --help Print help

Exemplo

Example

No exemplo a seguir, esse comando registra uma âncora de confiança no HSM. No máximo 2 (duas) âncoras de confiança podem ser registradas.

```
aws-cloudhsm > cluster mtls register-trust-anchor --path /home/rootCA
```

```
{
  "error_code": 0,
  "data": {
    "trust_anchor": {
      "certificate-reference": "0x01",
      "certificate": "<PEM Encoded Certificate>",
      "cluster-coverage": "full"
    }
  }
}
```

Em seguida, você pode executar o comando `list-trust-anchors` para confirmar que a âncora de confiança foi registrada no AWS CloudHSM:

```
aws-cloudhsm > cluster mtlb list-trust-anchors

{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
        "certificate": "<PEM Encoded Certificate>",
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PATH>

O caminho da âncora de confiança a ser registrada.

Obrigatório: sim

 Note

AWS CloudHSM suporta o registro de certificados intermediários como âncora confiável. Nesses casos, todo o arquivo da cadeia de certificados codificado pelo PEM precisa ser registrado no HSM, com os certificados em ordem hierárquica. AWS CloudHSM suporta uma cadeia de certificados de 6980 bytes.

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de cluster for maior que 1.

Tópicos relacionados

- [cluster mtls deregister-trust-anchor](#)
- [cluster mtls list-trust-anchors](#)
- [Configurar mTLS \(recomendado\)](#)

Definir o nível de imposição do mTLS com a CloudHSM CLI

Use o comando `cluster mtls set-enforcement` na CloudHSM CLI para definir o nível de imposição do uso do TLS mútuo entre o cliente e o AWS CloudHSM.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador com nome de usuário “admin”

Requisitos

Para executar esse comando:

- Pelo menos uma âncora de confiança foi registrada com êxito no AWS CloudHSM.
- Configure a CloudHSM CLI com a chave privada e o certificado de cliente corretos e inicie a CloudHSM CLI em uma conexão TLS mútua.

- Você deve fazer login como administrador padrão com o nome de usuário “admin”. Nenhum outro usuário administrador poderá executar esse comando.

Sintaxe

```
aws-cloudhsm > help cluster mtls set-enforcement
```

Set mtls enforcement policy in the cluster

Usage: cluster mtls set-enforcement [OPTIONS] --level [**<LEVEL>**...]

Options:

--cluster-id **<CLUSTER_ID>** Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--level **<LEVEL>** Level to be set for mtls in the cluster [possible values: none, cluster]

--approval **<APPROVAL>** Filepath of signed quorum token file to approve operation

-h, --help Print help

Exemplo

Example

No exemplo a seguir, esse comando define o nível de imposição mtls do cluster AWS CloudHSM a ser. O comando set-enforcement só pode ser executado em uma conexão TLS mútua estabelecida como usuário administrador com nome de usuário “admin”; consulte [definir a imposição do mTLS para AWS CloudHSM](#).

```
aws-cloudhsm > cluster mtls set-enforcement --level cluster
```

```
{
  "error_code": 0,
  "data": {
    "message": "Mtls enforcement level set to Cluster successfully"
  }
}
```

Em seguida, você pode executar o comando `get-enforcement` para confirmar se o nível de imposição foi definido como `cluster`:

```
aws-cloudhsm > cluster mtls get-enforcement

{
  "error_code": 0,
  "data": {
    "mtls-enforcement-level": "cluster"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<LEVEL>

Nível a ser definido para mTLS no cluster.

Valores válidos

- `cluster`: imponha o uso de TLS mútuo entre o cliente e AWS CloudHSM no cluster.
- `none`: não imponha o uso de TLS mútuo entre o cliente e AWS CloudHSM no cluster.

Obrigatório: sim

Warning

Depois de impor o uso de mTLS no cluster, todas as conexões não MTLS existentes serão descartadas e a conexão com o cluster será possível apenas com certificados mTLS.

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de cluster for maior que 1.

Tópicos relacionados

- [cluster mtls get-enforcement](#)
- [Configurar mTLS \(recomendado\)](#)

A categoria crypto na CLI do CloudHSM

Na CLI do CloudHSM, crypto é uma categoria principal para um grupo de comandos que, quando combinado com a categoria principal, cria um comando específico para operações criptográficas. Atualmente, essa categoria consiste nos seguintes comandos:

- [sign](#)
 - [ecdsa](#)
 - [ed25519ph](#)
 - [rsa-pkcs](#)
 - [rsa-pkcs-pss](#)
- [verificar](#)
 - [ecdsa](#)
 - [ed25519ph](#)
 - [rsa-pkcs](#)
 - [rsa-pkcs-pss](#)

A categoria de assinatura criptográfica na CLI do CloudHSM

Na CLI do CloudHSM, crypto sign é uma categoria principal de um grupo de comandos que, quando combinados com a categoria principal, usam uma chave privada escolhida no seu cluster do AWS CloudHSM para gerar uma assinatura. O comando crypto sign tem os seguintes subcomandos:

- [Gerar uma assinatura com o mecanismo ECDSA na CloudHSM CLI](#)
- [Gere uma assinatura com o mecanismo HashEd DSA na CLI do CloudHSM](#)
- [Gerar uma assinatura com o mecanismo RSA-PKCS na CloudHSM CLI](#)
- [Gere uma assinatura com o RSA-PKCS-PSS mecanismo na CLI do CloudHSM](#)

Para usar o crypto sign, primeiro você deve ter uma chave privada no HSM. Você pode gerar uma chave privada com os seguintes comandos:

- [chave, generate-asymmetric-pair etc.](#)
- [chave generate-asymmetric-pair rsa](#)

Gerar uma assinatura com o mecanismo ECDSA na CloudHSM CLI

Use o comando `crypto sign ecdsa` na CloudHSM CLI para gerar uma assinatura usando uma chave privada EC e o mecanismo de assinatura ECDSA.

Para usar o `crypto sign ecdsa` comando, primeiro você deve ter uma chave privada EC em seu AWS CloudHSM cluster. Você pode gerar uma chave privada EC usando o comando [Gerar um par de chaves EC assimétricas com a CloudHSM CLI](#) com o atributo `sign` definido como `true`.

A assinatura ECDSA resultante é gerada no formato `r | s`, em que os componentes `r` e `s` são concatenados como dados binários brutos e retornados no formato codificado em base64.

Note

As assinaturas podem ser verificadas AWS CloudHSM com [A categoria de verificação de criptografia na CLI do CloudHSM](#) subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help crypto sign ecdsa  
Sign with the ECDSA mechanism
```

```
Usage: crypto sign ecdsa --key-filter [<KEY_FILTER>] --hash-  
function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--key-filter [<KEY_FILTER>...]`

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key

`--hash-function <HASH_FUNCTION>`

[possible values: sha1, sha224, sha256, sha384, sha512]

`--data-path <DATA_PATH>`

The path to the file containing the data to be signed

`--data <DATA>`

Base64 Encoded data to be signed

`--approval <APPROVAL>`

Filepath of signed quorum token file to approve operation

`--data-type <DATA_TYPE>`

The type of data passed in, either raw or digest [possible values: raw, digest]

`-h, --help`

Print help

Exemplo

Esses exemplos mostram como usar o `crypto sign ecdsa` para gerar uma assinatura usando o mecanismo de assinatura ECDSA e a função hash SHA256. Esse comando usa uma chave privada no HSM.

Example Exemplo: gerar uma assinatura para dados codificados em base 64

```
aws-cloudhsm > crypto sign ecdsa --key-filter attr.label=ec-private --hash-function sha256 --data YWJjMTIz
{
  "error_code": 0,
  "data": {
    "key-reference": "0x000000000007808dd",
    "signature": "4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2w0S3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw=="
  }
}
```


Example Exemplo: gerar uma assinatura para um arquivo de dados

```
aws-cloudhsm > crypto sign ecdsa --key-filter attr.label=ec-private --hash-function sha256 --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000007808dd",
    "signature": "4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2w0S3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave correspondente.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte Atributos de chave da CloudHSM CLI.

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de uso da chave privada for maior que 1.

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Valores válidos:

- bruto
- resumo

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Gere uma assinatura com o mecanismo HashEd DSA na CLI do CloudHSM

Important

HashEdAs operações de assinatura de DSA só são suportadas em instâncias `hsm2m.medium` no modo não FIPS.

Use o `crypto sign ed25519ph` comando na CLI do CloudHSM para gerar uma assinatura usando uma chave privada Ed25519 e o mecanismo de assinatura do DSA. HashEd Para obter informações adicionais sobre o HashEd DSA, consulte [NIST SP 186-5](#), Seção 7.8.

Para usar o `crypto sign ed25519ph` comando, primeiro você deve ter uma chave privada Ed25519 em seu cluster. AWS CloudHSM Você pode gerar uma chave privada Ed25519 usando o [Gerar um par de chaves EC assimétricas com a CloudHSM CLI](#) comando com o `curve` parâmetro definido como `ed25519` e o `sign` atributo definido como `true`

Note

As assinaturas podem ser verificadas AWS CloudHSM com [A categoria de verificação de criptografia na CLI do CloudHSM](#) subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.
- HashEdAs operações de assinatura de DSA só são suportadas em instâncias `hsm2m.medium` no modo não FIPS.

Sintaxe

```
aws-cloudhsm > help crypto sign ed25519ph
```

Sign with the Ed25519ph mechanism

```
Usage: crypto sign ed25519ph [OPTIONS] --key-filter [<KEY_FILTER>...] --data-type <DATA_TYPE> --hash-function <HASH_FUNCTION> [--data-path <DATA_PATH>] --data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--key-filter [<KEY_FILTER>...]`

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key

`--data-path <DATA_PATH>`

The path to the file containing the data to be signed

`--data <DATA>`

Base64 Encoded data to be signed

`--approval <APPROVAL>`

Filepath of signed quorum token file to approve operation

`--data-type <DATA_TYPE>`

The type of data passed in, either raw or digest [possible values: raw, digest]

`--hash-function <HASH_FUNCTION>`

Hash function [possible values: sha512]

`-h, --help`

Print help

Exemplo

Esses exemplos mostram como usar crypto sign ed25519ph para gerar uma assinatura usando o mecanismo de assinatura ED25519ph e a função hash. sha512 Esse comando usa uma chave privada no HSM.

Example Exemplo: gerar uma assinatura para dados codificados em base 64

```
aws-cloudhsm > crypto sign ed25519ph \
--key-filter attr.label=ed25519-private \
--data-type raw \
--hash-function sha512 \
--data YWJj
{
  "error_code": 0,
  "data": {
    "key-reference": "0x00000000000401cdf",
    "signature": "mKcCIvC4Ehqp0w+BPWg/gJ5GK0acf/h20UmbuU5trkEx
+FBCRjwqNVogA9BirfWqoQuMYeY2Biqq0RwqJgg0Bg=="
  }
}
```

Example Exemplo: gerar uma assinatura para um arquivo de dados

```
aws-cloudhsm > crypto sign ed25519ph \
  --key-filter attr.label=ed25519-private \
  --data-type raw \
  --hash-function sha512 \
  --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x000000000000401cdf",
    "signature": "mKcCIvC4Ehqp0w+BPWg/gJ5GK0acf/h20UmbuU5trkEx
+FBCRjwqNVogA9BirfWqoQuMYeY2Biqq0RwqJgg0Bg=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: Sim (a menos que seja fornecido por meio do parâmetro de dados)

<HASH_FUNCTION>

Especifica a função hash. O ED25519ph suporta apenas. SHA512

Valores válidos:

- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave correspondente.

Para obter uma lista dos principais atributos da CLI do CloudHSM compatíveis, consulte.

[Atributos de chave da CloudHSM CLI](#)

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de uso da chave privada for maior que 1.

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Valores válidos:

- bruto
- resumo

Obrigatório: Sim

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Gerar uma assinatura com o mecanismo RSA-PKCS na CloudHSM CLI

Use o comando `crypto sign rsa-pkcs` na CloudHSM CLI para gerar uma assinatura usando uma chave privada RSA e o mecanismo de assinatura RSA-PKCS.

Para usar o `crypto sign rsa-pkcs` comando, primeiro você deve ter uma chave privada RSA em seu AWS CloudHSM cluster. Você pode gerar uma chave privada RSA usando o comando [Gerar um par de chaves assimétricas RSA com a CloudHSM CLI](#) com o atributo `sign` definido como `true`.

Note

As assinaturas podem ser verificadas AWS CloudHSM com [A categoria de verificação de criptografia na CLI do CloudHSM](#) subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help crypto sign rsa-pkcs
```

Sign with the RSA-PKCS mechanism

Usage: `crypto sign rsa-pkcs --key-filter [<KEY_FILTER>...] --hash-function <HASH_FUNCTION> [--data-path <DATA_PATH>|--data <DATA>]`

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--key-filter [<KEY_FILTER>...]`

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a matching key

`--hash-function <HASH_FUNCTION>`

[possible values: `sha1`, `sha224`, `sha256`, `sha384`, `sha512`]

`--data-path <DATA_PATH>`

The path to the file containing the data to be signed

```

--data <DATA>
    Base64 Encoded data to be signed
--approval <APPROVAL>
    Filepath of signed quorum token file to approve operation
--data-type <DATA_TYPE>
    The type of data passed in, either raw or digest [possible values: raw,
digest]
-h, --help
    Print help

```

Exemplo

Esses exemplos mostram como usar o `crypto sign rsa-pkcs` para gerar uma assinatura usando o mecanismo de assinatura RSA-PKCS e a função hash SHA256. Esse comando usa uma chave privada no HSM.

Example Exemplo: gerar uma assinatura para dados codificados em base 64

```

aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private --hash-function sha256 --data YWJjMTIz
{
  "error_code": 0,
  "data": {
    "key-reference": "0x000000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjg0QW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJ0BhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+G09BUT9FFU/
of9cvu40yn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3IT00qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ=="
  }
}

```

Example Exemplo: gerar uma assinatura para um arquivo de dados

```

aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private --hash-function sha256 --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x000000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjg0QW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJ0BhvGuM8J/

```



```

z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+G09BUT9FFU/
of9cvu40yn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEivFLG61sexJjQi8EdTxeDA
+I3IT00qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ=="
  }
}

```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio de dados)

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE selecionar uma chave correspondente.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte Atributos de chave da CloudHSM CLI.

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de uso da chave privada for maior que 1.

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Em relação a RSA-PKCS, os dados devem ser passados no formato codificado DER, conforme especificado na [RFC 8017, Seção 9.2](#)

Valores válidos:

- bruto
- resumo

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Gere uma assinatura com o RSA-PKCS-PSS mecanismo na CLI do CloudHSM

Use o comando `crypto sign rsa-pkcs-pss` na CloudHSM CLI para gerar uma assinatura usando uma chave privada RSA e o mecanismo de assinatura RSA-PKCS-PSS.

Para usar o `crypto sign rsa-pkcs-pss` comando, primeiro você deve ter uma chave privada RSA em seu AWS CloudHSM cluster. Você pode gerar uma chave privada RSA usando o comando [Gerar um par de chaves assimétricas RSA com a CloudHSM CLI](#) com o atributo `sign` definido como `true`.

Note

As assinaturas podem ser verificadas AWS CloudHSM com [A categoria de verificação de criptografia na CLI do CloudHSM](#) subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help crypto sign rsa-pkcs-pss
```

Sign with the RSA-PKCS-PSS mechanism

```
Usage: crypto sign rsa-pkcs-pss [OPTIONS] --key-filter [<KEY_FILTER>...] --
hash-function <HASH_FUNCTION> --mgf <MGF> --salt-length <SALT_LENGTH> <--data-
path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>          Unique Id to choose which of the clusters in the
config file to run the operation against. If not provided, will fall back to the value
provided when interactive mode was started, or error
--key-filter [<KEY_FILTER>...]      Key reference (e.g. key-
reference=0xabc) or space separated list of key attributes in the form of
attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key
--hash-function <HASH_FUNCTION>    [possible values: sha1, sha224, sha256, sha384,
sha512]
--data-path <DATA_PATH>            The path to the file containing the data to be
signed
--data <DATA>                      Base64 Encoded data to be signed
--mgf <MGF>                        The mask generation function [possible values:
mgf1-sha1, mgf1-sha224, mgf1-sha256, mgf1-sha384, mgf1-sha512]
--salt-length <SALT_LENGTH>        The salt length
```

<code>--approval <APPROVAL></code>	Filepath of signed quorum token file to approve operation
<code>--data-type <DATA_TYPE></code> [possible values: raw, digest]	The type of data passed in, either raw or digest
<code>-h, --help</code>	Print help

Exemplo

Esses exemplos mostram como usar o `crypto sign rsa-pkcs-pss` para gerar uma assinatura usando o mecanismo de assinatura RSA-PKCS-PSS e a função hash SHA256. Esse comando usa uma chave privada no HSM.

Example Exemplo: gerar uma assinatura para dados codificados em base 64

```
aws-cloudhsm > crypto sign rsa-pkcs-pss --key-filter attr.label=rsa-private --hash-function sha256 --data YWJjMTIz --salt-length 10 --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "key-reference": "0x000000000007008db",
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKV0Vm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRT7h/g4o6YERm1tTQL17q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7Yme0Ic1cZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemo0M16asSrEJSthaZWV530BsD0qzA8Rt8JdhXS+GZp3vNLdL10TBELDPweXVgAu4dBX0F0vpw/
gg6sNvuaDK4Y0Bv2fqKg=="
  }
}
```

Example Exemplo: gerar uma assinatura para um arquivo de dados

```
aws-cloudhsm > crypto sign rsa-pkcs-pss --key-filter attr.label=rsa-private --hash-function sha256 --data-path data.txt --salt-length 10 --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "key-reference": "0x000000000007008db",
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKV0Vm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRT7h/g4o6YERm1tTQL17q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7Yme0Ic1cZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemo0M16asSrEJSthaZWV530BsD0qzA8Rt8JdhXS+GZp3vNLdL10TBELDPweXVgAu4dBX0F0vpw/
gg6sNvuaDK4Y0Bv2fqKg=="
  }
}
```

```
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio de dados)

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave correspondente.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte Atributos de chave da CloudHSM CLI.

Obrigatório: Sim

<MGF>

Especifica a função de geração da máscara.

Note

A função hash da função de geração de máscara deve corresponder à função hash do mecanismo de assinatura.

Valores válidos:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obrigatório: Sim

<SALT_LENGTH>

Especifica o comprimento do sal.

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de uso da chave privada for maior que 1.

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Valores válidos:

- bruto
- resumo

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Tópicos relacionados

- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

A categoria de verificação de criptografia na CLI do CloudHSM

Na CLI do CloudHSM, `crypto verify` é uma categoria principal de um grupo de comandos que, quando combinados com a categoria principal, confirmam se um arquivo foi assinado por uma determinada chave. O comando `crypto verify` tem os seguintes subcomandos:

- [crypto verify ecdsa](#)
- [verificação criptográfica ed25519ph](#)
- [crypto verify rsa-pkcs](#)
- [verificação criptográfica rsa-pkcs-pss](#)

O comando `crypto verify` compara um arquivo assinado com um arquivo de origem e analisa se eles estão criptograficamente relacionados com base em uma determinada chave pública e no mecanismo de assinatura.

Note

Os arquivos podem ser conectados AWS CloudHSM com a [A categoria de assinatura criptográfica na CLI do CloudHSM](#) operação.

Verifique uma assinatura assinada com o mecanismo ECDSA na CloudHSM CLI

Use o comando `crypto verify ecdsa` na CloudHSM CLI para concluir as seguintes operações:

- Confirme se um arquivo foi assinado no HSM por uma determinada chave pública.
- Verifique se a assinatura foi gerada usando o mecanismo de assinatura ECDSA.
- Compare um arquivo assinado com um arquivo de origem e determina se os dois estão criptograficamente relacionados com base em uma determinada chave pública ecdsa e em um mecanismo de assinatura.
- A função de verificação do ECDSA espera a assinatura no formato `r | s`, em que os componentes `r` e `s` são concatenados como dados binários brutos.

Para usar o `crypto verify ecdsa` comando, primeiro você deve ter uma chave pública EC em seu AWS CloudHSM cluster. Você pode importar uma chave pública EC usando o comando [Importar uma chave no formato PEM com a CLI do CloudHSM](#) com o atributo `verify` definido como `true`.

Note

Você pode gerar uma assinatura na CloudHSM CLI com os subcomandos contidos em [A categoria de assinatura criptográfica na CLI do CloudHSM](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help crypto verify ecdsa
```

Verify with the ECDSA mechanism

```
Usage: crypto verify ecdsa --key-filter [<KEY_FILTER>...] --hash-  
function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>> <--signature-  
path <SIGNATURE_PATH>|--signature <SIGNATURE>>
```

Options:


```

--cluster-id <CLUSTER_ID>
    Unique Id to choose which of the clusters in the config file to run the
    operation against. If not provided, will fall back to the value provided when
    interactive mode was started, or error
--key-filter [<KEY_FILTER>...]
    Key reference (e.g. key-reference=0xabc) or space separated list of key
    attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
    matching key
--hash-function <HASH_FUNCTION>
    [possible values: sha1, sha224, sha256, sha384, sha512]
--data-path <DATA_PATH>
    The path to the file containing the data to be verified
--data <DATA>
    Base64 encoded data to be verified
--signature-path <SIGNATURE_PATH>
    The path to where the signature is located
--signature <SIGNATURE>
    Base64 encoded signature to be verified
--data-type <DATA_TYPE>
    The type of data passed in, either raw or digest [possible values: raw,
    digest]
-h, --help
    Print help

```

Exemplo

Esses exemplos mostram como usar o `crypto verify ecdsa` para verificar uma assinatura que foi gerada usando o mecanismo de assinatura ECDSA e a função hash SHA256. Esse comando usa uma chave pública no HSM.

Exemplo: verificar uma assinatura codificada em Base64 com dados codificados em Base64

```

aws-cloudhsm > crypto verify ecdsa --hash-function sha256 --key-filter attr.label=ec-
public --data YWJjMTIz --signature 4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2w0S3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}

```

Example Exemplo: verificar um arquivo de assinatura com um arquivo de dados

```
aws-cloudhsm > crypto verify ecdsa --hash-function sha256 --key-filter attr.label=ec-  
public --data-path data.txt --signature-path signature-file  
{  
  "error_code": 0,  
  "data": {  
    "message": "Signature verified successfully"  
  }  
}
```

Example Exemplo: provar relacionamento de assinatura falso

Esse comando verifica se os dados localizados em /home/data foram assinados por uma chave pública com o rótulo ecdsa-public usando o mecanismo de assinatura ECDSA para produzir a assinatura localizada em /home/signature. Como os argumentos fornecidos não formam um relacionamento de assinatura verdadeiro, o comando retorna uma mensagem de erro.

```
aws-cloudhsm > crypto verify ecdsa --hash-function sha256 --  
key-filter attr.label=ec-public --data aW52YWxpZA== --signature  
+ogk7M7S3iTqFg3SndJfd91dZFr5Qo6YixJl8JwcvqqVgsVu06o+VKvTRjz0/V05kf3JJbBLr87Q  
+wLWcMAJfA==  
{  
  "error_code": 1,  
  "data": "Signature verification failed"  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave correspondente.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte Atributos de chave da CloudHSM CLI.

Obrigatório: Sim

<SIGNATURE>

Assinatura codificada em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<SIGNATURE_PATH>

Especifica o local da assinatura.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Valores válidos:

- bruto
- resumo

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Verifique uma assinatura assinada com o mecanismo HashEd DSA na CLI do CloudHSM

 Important

HashEdAs operações de verificação de assinatura do DSA só são suportadas em instâncias hsm2m.medium no modo não FIPS.

Use o comando `crypto verify ed25519ph` na CloudHSM CLI para concluir as seguintes operações:

- Verifique as assinaturas de dados ou arquivos usando uma determinada chave pública Ed25519.
- Confirme se a assinatura foi gerada usando o mecanismo de assinatura HashEd DSA. Para obter informações adicionais sobre o HashEd DSA, consulte [NIST SP 186-5](#), Seção 7.8.

Para usar o `crypto verify ed25519ph` comando, primeiro você deve ter uma chave pública Ed25519 em seu cluster. AWS CloudHSM Você pode gerar um par de chaves Ed25519 usando o [Gerar um par de chaves EC assimétricas com a CloudHSM CLI](#) comando com o `curve` parâmetro definido como `ed25519` e o `verify` atributo definido como `true`, ou importar uma chave pública Ed25519 usando o [Importar uma chave no formato PEM com a CLI do CloudHSM](#) comando com o atributo definido como `.verify true`

 Note

Você pode gerar uma assinatura na CloudHSM CLI com os subcomandos contidos em [A categoria de assinatura criptográfica na CLI do CloudHSM](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.
- HashEdAs operações de verificação de assinatura do DSA só são suportadas em instâncias hsm2m.medium no modo não FIPS.

Sintaxe

```
aws-cloudhsm > help crypto verify ed25519ph
```

Verify with the Ed25519ph mechanism

Usage: `crypto verify ed25519ph [OPTIONS] --key-filter [<KEY_FILTER>...] --data-type <DATA_TYPE> --hash-function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>> <--signature-path <SIGNATURE_PATH>|--signature <SIGNATURE>>`

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--key-filter [<KEY_FILTER>...]`

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a matching key

`--data-path <DATA_PATH>`

The path to the file containing the data to be verified

`--data <DATA>`

Base64 encoded data to be verified

`--signature-path <SIGNATURE_PATH>`

The path to where the signature is located

`--signature <SIGNATURE>`

Base64 encoded signature to be verified

`--data-type <DATA_TYPE>`

The type of data passed in, either raw or digest [possible values: raw, digest]

```
--hash-function <HASH_FUNCTION>
    Hash function [possible values: sha512]
-h, --help
    Print help
```

Exemplo

Esses exemplos mostram como usar `crypto verify ed25519ph` para verificar uma assinatura que foi gerada usando o mecanismo de assinatura ED25519ph e a função hash. sha512. Esse comando usa uma chave pública Ed25519 no HSM.

Example Exemplo: verificar uma assinatura codificada em Base64 com dados codificados em Base64

```
aws-cloudhsm > crypto verify ed25519ph \
--hash-function sha512 \
--key-filter attr.label=ed25519-public \
--data-type raw \
--data YWJj \
--signature mKcCIvC4Ehqp0w+BPWg/gJ5GK0acf/h20UmbuU5trkEx
+FBCRjwqNVogA9BirfWqoQuMYeY2Biqq0RwqJgg0Bg==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Exemplo: verificar um arquivo de assinatura com um arquivo de dados

```
aws-cloudhsm > crypto verify ed25519ph \
--hash-function sha512 \
--key-filter attr.label=ed25519-public \
--data-type raw \
--data-path data.txt \
--signature-path signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem verificados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica a localização dos dados a serem verificados.

Obrigatório: Sim (a menos que seja fornecido por meio do parâmetro de dados)

<HASH_FUNCTION>

Especifica a função hash. O ED25519ph suporta apenas. SHA512

Valores válidos:

- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave correspondente.

Para obter uma lista dos principais atributos da CLI do CloudHSM compatíveis, consulte.

[Atributos de chave da CloudHSM CLI](#)

Obrigatório: Sim

<SIGNATURE>

Assinatura codificada em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<SIGNATURE_PATH>

Especifica o local da assinatura.

Obrigatório: Sim (a menos que seja fornecido por meio do parâmetro de assinatura)

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser codificado como parte do algoritmo de verificação. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Valores válidos:

- bruto
- resumo

Obrigatório: Sim

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)
- [Gere uma assinatura com o mecanismo HashEd DSA na CLI do CloudHSM](#)

Verificar uma assinatura assinada com o mecanismo RSA-PKCS na CloudHSM CLI

Use o comando `crypto verify rsa-pkcs` na CloudHSM CLI para concluir as seguintes operações:

- Confirme se um arquivo foi assinado no HSM por uma determinada chave pública.
- Verifique se a assinatura foi gerada usando o mecanismo de assinatura RSA-PKCS.
- Compare um arquivo assinado com um arquivo de origem e determina se os dois estão criptograficamente relacionados com base em uma determinada chave pública RSA e em um mecanismo de assinatura.

Para usar o `crypto verify rsa-pkcs` comando, primeiro você deve ter uma chave pública RSA em seu AWS CloudHSM cluster.

 Note

Você pode gerar uma assinatura usando a CloudHSM CLI com os subcomandos contidos em [A categoria de assinatura criptográfica na CLI do CloudHSM](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help crypto verify rsa-pkcs
```

Verify with the RSA-PKCS mechanism

Usage: `crypto verify rsa-pkcs --key-filter [<KEY_FILTER>...] --hash-function <HASH_FUNCTION> [--data-path <DATA_PATH>]|--data <DATA>> [--signature-path <SIGNATURE_PATH>]|--signature <SIGNATURE>>`

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--key-filter [<KEY_FILTER>...]`

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a matching key

`--hash-function <HASH_FUNCTION>`

[possible values: sha1, sha224, sha256, sha384, sha512]

`--data-path <DATA_PATH>`

The path to the file containing the data to be verified

`--data <DATA>`

Base64 encoded data to be verified

`--signature-path <SIGNATURE_PATH>`

```

    The path to where the signature is located
--signature <SIGNATURE>
    Base64 encoded signature to be verified
--data-type <DATA_TYPE>
    The type of data passed in, either raw or digest [possible values: raw,
digest]
-h, --help
    Print help

```

Exemplo

Esses exemplos mostram como usar `crypto verify rsa-pkcs` para verificar uma assinatura que foi gerada usando o mecanismo de assinatura RSA-PKCS e a função hash SHA256. Esse comando usa uma chave pública no HSM.

Example Exemplo: verificar uma assinatura codificada em Base64 com dados codificados em Base64

```

aws-cloudhsm > crypto verify rsa-pkcs --hash-function sha256 --key-filter
attr.label=rsa-public --data YWJjMTIz --signature XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjg0QW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBj0BhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+G09BUT9FFU/
of9cvu40yn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIVFLG61sexJjQi8EdTxeDA
+I3IT00qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}

```

Example Exemplo: verificar um arquivo de assinatura com um arquivo de dados

```

aws-cloudhsm > crypto verify rsa-pkcs --hash-function sha256 --key-filter
attr.label=rsa-public --data-path data.txt --signature-path signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}

```

Example Exemplo: provar relacionamento de assinatura falso

Esse comando verifica se os dados inválidos foram assinados por uma chave pública com o rótulo `rsa-public` usando o mecanismo de assinatura RSAPKCS para produzir a assinatura localizada em `/home/signature`. Como os argumentos fornecidos não formam um relacionamento de assinatura verdadeiro, o comando retorna uma mensagem de erro.

```
aws-cloudhsm > crypto verify rsa-pkcs --hash-function sha256 --key-filter
attr.label=rsa-public --data aW52YWxpZA== --signature XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjg0QW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBj0BhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+G09BUT9FFU/
of9cvu40yn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6ev1P7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEivFLG61sexJjQi8EdTxeDA
+I3IT00qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ==
{
  "error_code": 1,
  "data": "Signature verification failed"
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave correspondente.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte Atributos de chave da CloudHSM CLI.

Obrigatório: Sim

<SIGNATURE>

Assinatura codificada em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<SIGNATURE_PATH>

Especifica o local da assinatura.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Em relação a RSA-PKCS, os dados devem ser passados no formato codificado DER, conforme especificado na [RFC 8017, Seção 9.2](#)

Valores válidos:

- bruto
- resumo

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Verifique uma assinatura assinada com o RSA-PKCS-PSS mecanismo na CLI do CloudHSM

Use o comando `crypto sign rsa-pkcs-pss` na CloudHSM CLI para concluir as operações a seguir.

- Confirme se um arquivo foi assinado no HSM por uma determinada chave pública.
- Verifique se a assinatura foi gerada usando o mecanismo de assinatura RSA-PKCS-PSS.
- Compare um arquivo assinado com um arquivo de origem e determina se os dois estão criptograficamente relacionados com base em uma determinada chave pública RSA e em um mecanismo de assinatura.

Para usar o `crypto verify rsa-pkcs-pss` comando, primeiro você deve ter uma chave pública RSA em seu AWS CloudHSM cluster. Você pode importar uma chave pública RSA usando o comando `key import pem` (ADD UNWRAP LINK HERE) com o atributo `verify` definido como `true`.

Note

Você pode gerar uma assinatura usando a CloudHSM CLI com os subcomandos contidos em [A categoria de assinatura criptográfica na CLI do CloudHSM](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help crypto verify rsa-pkcs-pss
```

Verify with the RSA-PKCS-PSS mechanism

Usage: `crypto verify rsa-pkcs-pss --key-filter [<KEY_FILTER>...] --hash-function <HASH_FUNCTION> --mgf <MGF> --salt-length >SALT_LENGTH< <--data-path <DATA_PATH>|--data <DATA> <--signature-path <SIGNATURE_PATH>|--signature <SIGNATURE>>`

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--key-filter [<KEY_FILTER>...]`

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key

`--hash-function <HASH_FUNCTION>`

[possible values: sha1, sha224, sha256, sha384, sha512]

`--data-path <DATA_PATH>`

The path to the file containing the data to be verified

`--data <DATA>`

Base64 encoded data to be verified

`--signature-path <SIGNATURE_PATH>`

The path to where the signature is located

`--signature <SIGNATURE>`

Base64 encoded signature to be verified

`--data-type <DATA_TYPE>`

The type of data passed in, either raw or digest [possible values: raw, digest]

`--mgf <MGF>`

The mask generation function [possible values: mgf1-sha1, mgf1-sha224, mgf1-sha256, mgf1-sha384, mgf1-sha512]

`--salt-length <SALT_LENGTH>`

The salt length

`-h, --help`

Print help

Exemplo

Esses exemplos mostram como usar `crypto verify rsa-pkcs-pss` para verificar uma assinatura que foi gerada usando o mecanismo de RSA-PKCS-PSS assinatura e a função SHA256 hash. Esse comando usa uma chave pública no HSM.

Example Exemplo: verificar uma assinatura codificada em Base64 com dados codificados em Base64

```
aws-cloudhsm > crypto verify rsa-pkcs-pss --key-filter attr.label=rsa-public
--hash-function sha256 --data YWJjMTIz --salt-length 10 --mgf mgf1-sha256
--signature H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKV0Vm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBrt7h/g4o6YERm1tTQL17q+AJ7uGGK37zCsWQrAo7Vy8NzPSHxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7Yme0Ic1cZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjPN
+m4FNUds30GAemo0M16asSrEJSthaZWV530BsD0qzA8Rt8JdhXS+GZp3vNLdL10TBELDPweXVgAu4dBX0F0vpw/
gg6sNvuaDK4Y0Bv2fqKg==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Exemplo: verificar um arquivo de assinatura com um arquivo de dados

```
aws-cloudhsm > crypto verify rsa-pkcs-pss --key-filter attr.label=rsa-public --hash-
function sha256 --data-path data.txt --salt-length 10 --mgf mgf1-sha256 --signature
signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Exemplo: provar relacionamento de assinatura falso

Esse comando verifica se os dados inválidos foram assinados por uma chave pública com o rótulo `rsa-public` usando o mecanismo de assinatura RSAPKCSPSS para produzir a assinatura localizada em `/home/signature`. Como os argumentos fornecidos não formam um relacionamento de assinatura verdadeiro, o comando retorna uma mensagem de erro.

```
aws-cloudhsm > crypto verify rsa-pkcs-pss --key-filter attr.label=rsa-public
--hash-function sha256 --data aW52YWxpZA== --salt-length 10 --mgf mgf1-sha256
--signature H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKV0Vm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBrt7h/g4o6YERm1tTQL17q+AJ7uGGK37zCsWQrAo7Vy8NzPSHxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7Yme0Ic1cZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjPN
```

```
+m4FNUds30GAemo0M16asSrEJSthaZWV530BsD0qzA8Rt8JdhXS+GZp3vNLdL10TBELDPweXVgAu4dBX0F0vpw/  
gg6sNvuaDK4Y0Bv2fqKg==  
{  
  "error_code": 1,  
  "data": "Signature verification failed"  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<DATA>

Dados codificados em Base64 a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<DATA_PATH>

Especifica o local dos dados a serem assinados.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<KEY_FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE selecionar uma chave correspondente.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte Atributos de chave da CloudHSM CLI.

Obrigatório: Sim

<MFG>

Especifica a função de geração da máscara.

Note

A função hash da função de geração de máscara deve corresponder à função hash do mecanismo de assinatura.

Valores válidos:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obrigatório: Sim

<SIGNATURE>

Assinatura codificada em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<SIGNATURE_PATH>

Especifica o local da assinatura.

Obrigatório: sim (a menos que seja fornecido por meio do caminho da assinatura)

<DATA_TYPE>

Especifica se o valor do parâmetro de dados deve ser criptografado com hash como parte do algoritmo de assinatura. Use `raw` para dados sem hash; use `digest` para resumos, que já estão criptografados com hash.

Valores válidos:

- bruto
- resumo

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

A key na CLI do CloudHSM

Na CLI do CloudHSM, key é uma categoria principal para um grupo de comandos que, quando combinado com a categoria principal, cria um comando específico para chaves. Atualmente, essa categoria consiste nos seguintes comandos:

- [excluir](#)
- [generate-file](#)
- [chave generate-asymmetric-pair](#)
 - [chave generate-asymmetric-pair rsa](#)
 - [chave, generate-asymmetric-pair etc.](#)
- [key generate-symmetric](#)
 - [key generate-symmetric aes](#)
 - [key generate-symmetric generic-secret](#)
- [importar pem](#)
- [listar](#)
- [replicar](#)
- [set-attribute](#)
- [compartilhar](#)
- [cancelar compartilhamento](#)
- [unwrap](#)
- [wrap](#)

Excluir uma chave com a CLI do CloudHSM

Use o `key delete` comando na CLI do CloudHSM para excluir uma chave de um cluster. AWS CloudHSM Você só pode excluir uma chave por vez. A exclusão de uma chave em um par de chaves não tem efeito na outra chave do par.

Somente o CU que criou a chave e, conseqüentemente, a possui pode excluir a chave. Os usuários que compartilham a chave, mas não a possuem, podem usá-la em operações criptográficas, mas não podem excluí-la.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key delete  
Delete a key in the HSM cluster
```

```
Usage: key delete [OPTIONS] --filter [<FILTER>...]
```

Options:

```
--cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the  
config file to run the operation against. If not provided, will fall back to the value  
provided when interactive mode was started, or error  
--filter [<FILTER>...] Key reference (e.g. key-reference=0xabc)  
or space separated list of key attributes in the form of  
attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key for deletion  
-h, --help Print help
```

Exemplo

```
aws-cloudhsm > key delete --filter attr.label="ec-test-public-key"  
{
```

```
"error_code": 0,  
"data": {  
  "message": "Key deleted successfully"  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave da CLI do CloudHSM com suporte, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: Sim

Tópicos relacionados

- [Listar chaves para um usuário com a CloudHSM CLI](#)
- [Exportar uma chave assimétrica com a CloudHSM CLI](#)
- [Cancelar o compartilhamento de uma chave usando a CloudHSM CLI](#)
- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

Exportar uma chave assimétrica com a CloudHSM CLI

Use o comando key generate-file na CloudHSM CLI para exportar uma chave assimétrica do módulo de segurança de hardware (HSM). Se o destino for uma chave privada, a referência à chave privada será exportada no formato PEM falso. Se o destino for uma chave pública, os bytes da chave pública serão exportados no formato PEM.

O arquivo PEM falso, que não contém o material real da chave privada, mas faz referência à chave privada no HSM, pode ser usado para estabelecer o SSL/TLS descarregamento do seu servidor web para o. AWS CloudHSM Para obter mais informações, consulte [Descarregamento de SSL/TLS](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key generate-file
```

Generate a key file from a key in the HSM cluster. This command does not export any private key data from the HSM

Usage: key generate-file --encoding **<ENCODING>** --path **<PATH>** --filter [**<FILTER>**...]

Options:

--cluster-id **<CLUSTER_ID>**

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--encoding **<ENCODING>**

Encoding format for the key file

Possible values:

- reference-pem: PEM formatted key reference (supports private keys)
- pem: PEM format (supports public keys)

--path **<PATH>**

Filepath where the key file will be written

--filter [<FILTER>**...]**

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key for file generation

-h, --help

Print help (see a summary with '-h')

Exemplo

Este exemplo mostra como usar `key generate-file` para gerar um arquivo de chave em seu AWS CloudHSM cluster.

Example

```
aws-cloudhsm > key generate-file --encoding reference-pem --path /tmp/ec-private-  
key.pem --filter attr.label="ec-test-private-key"  
{  
  "error_code": 0,  
  "data": {  
    "message": "Successfully generated key file"  
  }  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte [Atributos de chave da CloudHSM CLI](#)

Obrigatório: não

<ENCODING>

Especifica o formato de codificação para o arquivo de chave

Obrigatório: Sim

<PATH>

Especifica o caminho do arquivo em que o arquivo de chave será gravado

Obrigatório: Sim

Gerar referências de chave KSP (Windows)**Note**

Esse atributo só está disponível no SDK versão 5.16.0 e posterior.

Pré-requisitos

- Você pode gerar referências de chave KSP somente em plataformas Windows.
- Você deve fazer login como usuário de criptografia (CU).

Local do arquivo

Por padrão, o AWS CloudHSM armazena arquivos gerados em: C:\Users\Default\AppData \Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition

Para especificar uma localização diferente, use o parâmetro --path.

Sintaxe

```
aws-cloudhsm > help key generate-file --encoding ksp-key-reference
```

Generate a key file from a key in the HSM cluster. This command does not export any private key data from the HSM

Usage: key generate-file --encoding **<ENCODING>** --path **<PATH>** --filter [**<FILTER>**...]

Options:

--encoding **<ENCODING>**

Encoding format for the key file

Possible values:

- reference-pem: PEM formatted key reference (supports private keys)
- pem: PEM format (supports public keys)
- ksp-key-reference: KSP key reference format

```

--cluster-id <CLUSTER_ID>
    Unique Id to choose which of the clusters in the config file to run the
    operation against. If not provided with multiple clusters configured, will error

--path <PATH>
    Directory path where the key file will be written

--filter [<FILTER>...]
    Key reference (e.g. key-reference=0xabc) or space separated list of key
    attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
    matching key for file generation

--all
    Generate ksp key reference for all available key pairs in HSM

-h, --help
    Print help (see a summary with '-h')
```

Exemplo: gerar uma referência de chave KSP usando um filtro de atributos de uma chave privada

O exemplo a seguir gera uma referência de chave KSP para uma chave privada com um rótulo específico.

Example

```

aws-cloudhsm > key generate-file --encoding ksp-key-reference --path --filter
attr.label="ec-test-private-key"
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Exemplo: gerar referências de chave KSP para todos os pares de chaves

O exemplo a seguir gera referências de chave KSP para todos os pares de chaves em seu cluster.

Example

```

aws-cloudhsm > key generate-file --encoding ksp-key-reference --all
{
```



```
"error_code": 0,  
"data": {  
  "message": "Successfully generated key file"  
}  
}
```

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)
- [A generate-asymmetric-pair categoria na CLI do CloudHSM](#)
- [A categoria generate-symmetric na CLI do CloudHSM](#)

A generate-asymmetric-pair categoria na CLI do CloudHSM

Na CLI do CloudHSM, key generate-asymmetric-pair é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando que gera pares de chaves assimétricas. Atualmente, essa categoria consiste nos seguintes comandos:

- [chave generate-asymmetric-pair etc.](#)
- [chave generate-asymmetric-pair rsa](#)

Gerar um par de chaves EC assimétricas com a CloudHSM CLI

Use o key asymmetric-pair ec comando na CLI do CloudHSM para gerar um par de chaves de curva elíptica (EC) assimétrica em seu cluster. AWS CloudHSM

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key generate-asymmetric-pair ec
```

Generate an Elliptic-Curve Cryptography (ECC) key pair

Usage: `key generate-asymmetric-pair ec [OPTIONS] --public-label <PUBLIC_LABEL> --private-label <PRIVATE_LABEL> --curve <CURVE>`

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--public-label <PUBLIC_LABEL>`

Label for the public key

`--private-label <PRIVATE_LABEL>`

Label for the private key

`--session`

Creates a session key pair that exists only in the current session. The key cannot be recovered after the session ends

`--curve <CURVE>`

Elliptic curve used to generate the key pair [possible values: prime256v1, secp256r1, secp224r1, secp384r1, secp256k1, secp521r1, ed25519]

`--public-attributes [<PUBLIC_KEY_ATTRIBUTES>...]`

Space separated list of key attributes to set for the generated EC public key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE

`--private-attributes [<PRIVATE_KEY_ATTRIBUTES>...]`

Space separated list of key attributes to set for the generated EC private key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE

`--share-crypto-users [<SHARE_CRYPTO_USERS>...]`

Space separated list of Crypto User usernames to share the EC private key with

`--manage-private-key-quorum-value <MANAGE_PRIVATE_KEY_QUORUM_VALUE>`

The quorum value for key management operations for the private key

`--use-private-key-quorum-value <USE_PRIVATE_KEY_QUORUM_VALUE>`

The quorum value for key usage operations for the private key

`-h, --help`

Print help

Exemplos

Esses exemplos mostram como usar o `key generate-asymmetric-pair ec` comando para criar um par de chaves EC.

Example Exemplo: criar um par de chaves EC

```
aws-cloudhsm > key generate-asymmetric-pair ec \
```

```

--curve secp224r1 \
--public-label ec-public-key-example \
--private-label ec-private-key-example
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000012000b",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",
        "id": "",
        "check-value": "0xd7c1a7",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,

```

```

    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 57,
    "ec-point":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f38a
    "curve": "secp224r1"
  }
},
"private_key": {
  "key-reference": "0x000000000012000c",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "ec",
    "label": "ec-private-key-example",
    "id": "",
    "check-value": "0xd7c1a7",
    "class": "private-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": false,
    "trusted": false,

```

```

    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 122,
    "ec-point":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f38a
    "curve": "secp224r1"
  }
}
}
}
}

```

Example Exemplo: criar um par de chaves EC com atributos opcionais

```

aws-cloudhsm > key generate-asymmetric-pair ec \
  --curve secp224r1 \
  --public-label ec-public-key-example \
  --private-label ec-private-key-example \
  --public-attributes encrypt=true \
  --private-attributes decrypt=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000002806eb",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",

```

```

    "id": "",
    "check-value": "0xedef86",
    "class": "public-key",
    "encrypt": true,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 57,
    "ec-point":
"0x00487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
    "curve": "secp224r1"
  }
},
"private_key": {
  "key-reference": "0x00000000000280c82",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {

```

```

    "key-type": "ec",
    "label": "ec-private-key-example",
    "id": "",
    "check-value": "0xedef86",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 122,
    "ec-point":
      "0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
      "curve": "secp224r1"
  }
}
}
}

```

Example Exemplo: criar um par de chaves EC com valores de quórum

Ao gerar uma chave com controles de quórum, a chave deve estar associada a um número mínimo de usuários igual ao maior valor de quórum da chave. Os usuários associados incluem o proprietário da chave e os usuários de criptografia com quem essa chave é compartilhada. Para definir o número mínimo de usuários com quem compartilhar a chave, obtenha o maior valor de quórum entre o valor do quórum de uso da chave e o valor do quórum de gerenciamento de chaves e subtraia 1 para contabilizar o proprietário da chave, que, por padrão, está associado à chave. Para compartilhar a chave com mais usuários, use o comando [Compartilhar uma chave usando a CloudHSM CLI](#).

```
aws-cloudhsm > key generate-asymmetric-pair ec \
  --curve secp224r1 \
  --public-label ec-public-key-example \
  --private-label ec-private-key-example \
  --public-attributes verify=true \
  --private-attributes sign=true
  --share-crypto-users cu2 cu3 cu4 \
  --manage-private-key-quorum-value 4 \
  --use-private-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000002806eb",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",
        "id": "",
        "check-value": "0xedef86",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,

```



```

    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 57,
    "ec-point":
"0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
    "curve": "secp224r1"
  }
},
"private_key": {
  "key-reference": "0x00000000000280c82",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [
      {
        "username": "cu2",
        "key-coverage": "full"
      },
      {
        "username": "cu3",
        "key-coverage": "full"
      },
      {
        "username": "cu4",
        "key-coverage": "full"
      }
    ],
    "key-quorum-values": {
      "manage-key-quorum-value": 4,
      "use-key-quorum-value": 2
    },
    "cluster-coverage": "full"
  },
},

```

```

    "attributes": {
      "key-type": "ec",
      "label": "ec-private-key-example",
      "id": "",
      "check-value": "0xedef86",
      "class": "private-key",
      "encrypt": false,
      "decrypt": false,
      "token": true,
      "always-sensitive": true,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": true,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": true,
      "trusted": false,
      "unwrap": false,
      "verify": false,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 122,
      "ec-point":
"0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
      "curve": "secp224r1"
    }
  }
}
}

```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<CURVE>

Especifica o identificador da curva elíptica.

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1
- ed25519 (compatível somente em instâncias hsm2m.medium no modo não FIPS)

Obrigatório: Sim

<PUBLIC_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos principais atributos a serem definidos para a chave pública EC gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, verify=true)

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<PUBLIC_LABEL>

Especifica um rótulo definido pelo usuário para a chave pública. O tamanho máximo permitido para label é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<PRIVATE_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos principais atributos a serem definidos para a chave privada EC gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true)

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário. O tamanho máximo permitido para `label` é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Por padrão, as chaves geradas são chaves persistentes (tokens). Passar `<SESSION>` muda isso, garantindo que uma chave gerada com esse argumento seja uma chave de sessão (efêmera).

Obrigatório: não

<SHARE_CRYPTO_USERS>

Especifica uma lista separada por espaços de nomes de usuário dos Usuários de criptografia com os quais compartilhar a chave privada EC

Obrigatório: não

<MANAGE_PRIVATE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de gerenciamento da chave privada. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

<USE_PRIVATE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de uso da chave privada. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

Gerar um par de chaves assimétricas RSA com a CloudHSM CLI

Use o `key generate-asymmetric-pair rsa` comando na CLI do CloudHSM para gerar um par de chaves RSA assimétrico em seu cluster. AWS CloudHSM

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key generate-asymmetric-pair rsa
```

Generate an RSA key pair

Usage: `key generate-asymmetric-pair rsa [OPTIONS] --public-label <PUBLIC_LABEL> --private-label <PRIVATE_LABEL> --modulus-size-bits <MODULUS_SIZE_BITS> --public-exponent <PUBLIC_EXPONENT>`

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--public-label <PUBLIC_LABEL>`

Label for the public key

`--private-label <PRIVATE_LABEL>`

Label for the private key

`--session`

Creates a session key pair that exists only in the current session. The key cannot be recovered after the session ends

`--modulus-size-bits <MODULUS_SIZE_BITS>`

```

    Modulus size in bits used to generate the RSA key pair
--public-exponent <PUBLIC_EXPONENT>
    Public exponent used to generate the RSA key pair
--public-attributes [<PUBLIC_KEY_ATTRIBUTES>...]
    Space separated list of key attributes to set for the generated RSA public
key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
--private-attributes [<PRIVATE_KEY_ATTRIBUTES>...]
    Space separated list of key attributes to set for the generated RSA private
key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
--share-crypto-users [<SHARE_CRYPTO_USERS>...]
    Space separated list of Crypto User usernames to share the RSA key with
--manage-private-key-quorum-value <MANAGE_PRIVATE_KEY_QUORUM_VALUE>
    The quorum value for key management operations for the private key
--use-private-key-quorum-value <USE_PRIVATE_KEY_QUORUM_VALUE>
    The quorum value for key usage operations for the private key
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar `key generate-asymmetric-pair rsa` para criar um par de chaves RSA.

Example Exemplo: criar um par de chaves RSA

```

aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000000160010",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ]
      },
      "shared-users": [],

```

```

    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "rsa",
    "label": "rsa-public-key-example",
    "id": "",
    "check-value": "0x498e1f",
    "class": "public-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
      "0xdfca0669dc8288ed3bad99509bd21c7e6192661407021b3f4cdf4a593d939dd24f4d641af8e4e73b04c847731c6
      e89a065e7d1a46ced96b46b909db2ab6be871ee700fd0a448b6e975bb64cae77c49008749212463e37a577baa57ce3e
      bcebb7d20bd6df1948ae336ae23b52d73b7f3b6acc2543edb6358e08d326d280ce489571f4d34e316a2ea1904d513ca
      "modulus-size-bits": 2048
  }
},
"private_key": {
  "key-reference": "0x0000000000160011",
  "key-info": {
    "key-owners": [
      {

```

```

        "username": "cu1",
        "key-coverage": "full"
    }
],
"shared-users": [],
"key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
},
"cluster-coverage": "full"
},
"attributes": {
    "key-type": "rsa",
    "label": "rsa-private-key-example",
    "id": "",
    "check-value": "0x498e1f",
    "class": "private-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1217,
    "public-exponent": "0x010001",
    "modulus":
"0xdfca0669dc8288ed3bad99509bd21c7e6192661407021b3f4cdf4a593d939dd24f4d641af8e4e73b04c847731c6
    "modulus-size-bits": 2048
}
}
}

```



```
}
```

Example Exemplo: criar um par de chaves RSA com atributos opcionais

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes encrypt=true \
--private-attributes decrypt=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
```

```

    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
      "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
      73a80fdb457aa7b20cd61e486c326e2cfd5e124a7f6a996437437812b542e3caf85928aa866f0298580f7967ee6aa01
      f6e6296d6c116d5744c6d60d14d3bf3cb978fe6b75ac67b7089bafd50d8687213b31abc7dc1bad422780d29c851d510
      133022653225bd129f8491101725e9ea33e1ded83fb57af35f847e532eb30cd7e726f23910d2671c6364092e834697e
      ac3160f0ca9725d38318b7",
    "modulus-size-bits": 2048
  },
  "private_key": {
    "key-reference": "0x00000000000280cc7",
    "key-info": {
      "key-owners": [
        {
          "username": "cu1",
          "key-coverage": "full"
        }
      ],
      "shared-users": [],
      "key-quorum-values": {
        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
      },
      "cluster-coverage": "full"
    },
    "attributes": {
      "key-type": "rsa",
      "label": "rsa-private-key-example",
      "id": "",
      "check-value": "0x01fe6e",
      "class": "private-key",

```

```

    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1217,
    "public-exponent": "0x010001",
    "modulus":
      "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
      "modulus-size-bits": 2048
  }
}
}
}
}

```

Example Exemplo: criar um par de chaves RSA com valores de quórum

Ao gerar uma chave com controles de quórum, a chave deve estar associada a um número mínimo de usuários igual ao maior valor de quórum da chave. Os usuários associados incluem o proprietário da chave e os usuários de criptografia com quem essa chave é compartilhada. Para definir o número mínimo de usuários com quem compartilhar a chave, obtenha o maior valor de quórum entre o valor do quórum de uso da chave e o valor do quórum de gerenciamento de chaves e subtraia 1 para contabilizar o proprietário da chave, que, por padrão, está associado à chave. Para compartilhar a chave com mais usuários, use o comando [Compartilhar uma chave usando a CloudHSM CLI](#).

```

aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \

```

```
--private-label rsa-private-key-example \  
--public-attributes verify=true \  
--private-attributes sign=true \  
--share-crypto-users cu2 cu3 cu4 \  
--manage-private-key-quorum-value 4 \  
--use-private-key-quorum-value 2  
{  
  "error_code": 0,  
  "data": {  
    "public_key": {  
      "key-reference": "0x00000000000280cc8",  
      "key-info": {  
        "key-owners": [  
          {  
            "username": "cu1",  
            "key-coverage": "full"  
          }  
        ],  
        "shared-users": [],  
        "key-quorum-values": {  
          "manage-key-quorum-value": 0,  
          "use-key-quorum-value": 0  
        },  
        "cluster-coverage": "full"  
      },  
      "attributes": {  
        "key-type": "rsa",  
        "label": "rsa-public-key-example",  
        "id": "",  
        "check-value": "0x01fe6e",  
        "class": "public-key",  
        "encrypt": false,  
        "decrypt": false,  
        "token": true,  
        "always-sensitive": false,  
        "derive": false,  
        "destroyable": true,  
        "extractable": true,  
        "local": true,  
        "modifiable": true,  
        "never-extractable": false,  
        "private": true,  
        "sensitive": false,  
        "sign": true,
```

```

    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
      "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
      73a80fdb457aa7b20cd61e486c326e2cfd5e124a7f6a996437437812b542e3caf85928aa866f0298580f7967ee6aa01
      f6e6296d6c116d5744c6d60d14d3bf3cb978fe6b75ac67b7089bafd50d8687213b31abc7dc1bad422780d29c851d510
      133022653225bd129f8491101725e9ea33e1ded83fb57af35f847e532eb30cd7e726f23910d2671c6364092e834697e
      ac3160f0ca9725d38318b7",
    "modulus-size-bits": 2048
  }
},
"private_key": {
  "key-reference": "0x00000000000280cc7",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [
      {
        "username": "cu2",
        "key-coverage": "full"
      },
      {
        "username": "cu3",
        "key-coverage": "full"
      },
      {
        "username": "cu4",
        "key-coverage": "full"
      }
    ],
    "key-quorum-values": {
      "manage-key-quorum-value": 4,
      "use-key-quorum-value": 2
    },
    "cluster-coverage": "full"
  }
}

```

```

    },
    "attributes": {
      "key-type": "rsa",
      "label": "rsa-private-key-example",
      "id": "",
      "check-value": "0x01fe6e",
      "class": "private-key",
      "encrypt": false,
      "decrypt": false,
      "token": true,
      "always-sensitive": true,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": true,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": true,
      "trusted": false,
      "unwrap": false,
      "verify": false,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 1217,
      "public-exponent": "0x010001",
      "modulus":
"0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
      "modulus-size-bits": 2048
    }
  }
}
}

```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<MODULUS_SIZE_BITS>

Especifica o comprimento do módulo em bits. O valor mínimo é 2048.

Obrigatório: Sim

<PRIVATE_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave privada RSA gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true)

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário. O tamanho máximo permitido para label é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<PUBLIC_EXPONENT>

Especifica o expoente público. O valor deve ser um número ímpar maior que ou igual a 65537.

Obrigatório: Sim

<PUBLIC_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave pública RSA gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, verify=true)

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<PUBLIC_LABEL>

Especifica um rótulo definido pelo usuário para a chave pública. O tamanho máximo permitido para `label` é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Por padrão, as chaves geradas são chaves persistentes (tokens). Passar `<SESSION>` muda isso, garantindo que uma chave gerada com esse argumento seja uma chave de sessão (efêmera).

Obrigatório: não

<SHARE_CRYPTO_USERS>

Especifica uma lista separada por espaços de nomes de usuário dos Usuários de criptografia com os quais compartilhar a chave privada RSA

Obrigatório: não

<MANAGE_PRIVATE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de gerenciamento da chave privada. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

<USE_PRIVATE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de uso da chave privada. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

A categoria generate-symmetric na CLI do CloudHSM

Na CLI do CloudHSM, `key generate-symmetric` é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando que gera chaves simétricas. Atualmente, essa categoria consiste nos seguintes comandos:

- [key generate-symmetric aes](#)
- [key generate-symmetric generic-secret](#)

Gerar uma chave AES simétrica com a CloudHSM CLI

Use o `key generate-symmetric aes` comando na CLI do CloudHSM para gerar uma chave AES simétrica no seu cluster. AWS CloudHSM

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key generate-symmetric aes
```

```
Generate an AES key
```

```
Usage: key generate-symmetric aes [OPTIONS] --label <LABEL> --key-length-  
bytes <KEY_LENGTH_BYTES>
```

Options:

```

--cluster-id <CLUSTER_ID>
    Unique Id to choose which of the clusters in the config file to run the
    operation against. If not provided, will fall back to the value provided when
    interactive mode was started, or error
--label <LABEL>
    Label for the key
--session
    Creates a session key that exists only in the current session. The key cannot
    be recovered after the session ends
--key-length-bytes <KEY_LENGTH_BYTES>
    Key length in bytes
--attributes [<KEY_ATTRIBUTES>...]
    Space separated list of key attributes to set for the generated AES key in
    the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
--share-crypto-users [<SHARE_CRYPTO_USERS>...]
    Space separated list of Crypto User usernames to share the AES key with
--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>
    The quorum value for key management operations
--use-key-quorum-value <USE_KEY_QUORUM_VALUE>
    The quorum value for key usage operations
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar o `key generate-symmetric aes` comando para criar uma chave AES.

Example Exemplo: crie uma chave AES

```

aws-cloudhsm > key generate-symmetric aes \
--label example-aes \
--key-length-bytes 24
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x0000000000002e06bf",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",

```

```

        "key-coverage": "full"
    },
    ],
    "shared-users": [],
    "key-quorum-values": {
        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
},
"attributes": {
    "key-type": "aes",
    "label": "example-aes",
    "id": "",
    "check-value": "0x9b94bd",
    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 24
}
}
}
}

```

Example Exemplo: crie uma chave AES com atributos opcionais

```
aws-cloudhsm > key generate-symmetric aes \
```

```
--label example-aes \  
--key-length-bytes 24 \  
--attributes decrypt=true encrypt=true  
{  
  "error_code": 0,  
  "data": {  
    "key": {  
      "key-reference": "0x000000000002e06bf",  
      "key-info": {  
        "key-owners": [  
          {  
            "username": "cu1",  
            "key-coverage": "full"  
          }  
        ],  
        "shared-users": [],  
        "key-quorum-values": {  
          "manage-key-quorum-value": 0,  
          "use-key-quorum-value": 0  
        },  
        "cluster-coverage": "full"  
      },  
      "attributes": {  
        "key-type": "aes",  
        "label": "example-aes",  
        "id": "",  
        "check-value": "0x9b94bd",  
        "class": "secret-key",  
        "encrypt": true,  
        "decrypt": true,  
        "token": true,  
        "always-sensitive": true,  
        "derive": false,  
        "destroyable": true,  
        "extractable": true,  
        "local": true,  
        "modifiable": true,  
        "never-extractable": false,  
        "private": true,  
        "sensitive": true,  
        "sign": true,  
        "trusted": false,  
        "unwrap": false,  
        "verify": true,
```

```

        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 24
    }
}
}
}

```

Example Exemplo: criar uma chave AES com valores de quórum

Ao gerar uma chave com controles de quórum, a chave deve estar associada a um número mínimo de usuários igual ao maior valor de quórum da chave. Os usuários associados incluem o proprietário da chave e os usuários de criptografia com quem essa chave é compartilhada. Para definir o número mínimo de usuários com quem compartilhar a chave, obtenha o maior valor de quórum entre o valor do quórum de uso da chave e o valor do quórum de gerenciamento de chaves e subtraia 1 para contabilizar o proprietário da chave, que, por padrão, está associado à chave. Para compartilhar a chave com mais usuários, use o comando [Compartilhar uma chave usando a CloudHSM CLI](#).

```

aws-cloudhsm > key generate-symmetric aes \
--label example-aes \
--key-length-bytes 24 \
--attributes decrypt=true encrypt=true
--share-crypto-users cu2 cu3 cu4 \
--manage-key-quorum-value 4 \
--use-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000002e06bf",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          }
        ]
      }
    }
  }
}

```

```
{
  "username": "cu3",
  "key-coverage": "full"
},
{
  "username": "cu4",
  "key-coverage": "full"
},
],
"key-quorum-values": {
  "manage-key-quorum-value": 4,
  "use-key-quorum-value": 2
},
"cluster-coverage": "full"
},
"attributes": {
  "key-type": "aes",
  "label": "example-aes",
  "id": "",
  "check-value": "0x9b94bd",
  "class": "secret-key",
  "encrypt": true,
  "decrypt": true,
  "token": true,
  "always-sensitive": true,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": true,
  "modifiable": true,
  "never-extractable": false,
  "private": true,
  "sensitive": true,
  "sign": true,
  "trusted": false,
  "unwrap": false,
  "verify": true,
  "wrap": false,
  "wrap-with-trusted": false,
  "key-length-bytes": 24
}
}
}
```

```
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave AES gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true).

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<KEY-LENGTH-BYTES>

Especifica o tamanho da chave em bytes.

Valores válidos:

- 16, 24 e 32

Obrigatório: Sim

<LABEL>

Especifica o rótulo da chave privada definida pelo usuário para a chave AES. O tamanho máximo permitido para label é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Por padrão, as chaves geradas são chaves persistentes (tokens). Passar <SESSION> muda isso, garantindo que uma chave gerada com esse argumento seja uma chave de sessão (efêmera).

Obrigatório: não

<SHARE_CRYPTO_USERS>

Especifica uma lista separada por espaços de nomes de usuário dos Usuários de criptografia com os quais compartilhar a chave AES

Obrigatório: não

<MANAGE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de gerenciamento da chave. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

<USE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de uso da chave. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

Gerar uma chave secreta genérica simétrica com a CloudHSM CLI

Use o `key generate-asymmetric-pair` comando na CLI do CloudHSM para gerar uma chave secreta genérica simétrica em seu cluster. AWS CloudHSM

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > key help generate-symmetric generic-secret
```

Generate a generic secret key

Usage: key generate-symmetric generic-secret [OPTIONS] --label **<LABEL>** --key-length-bytes **<KEY_LENGTH_BYTES>**

Options:

--cluster-id **<CLUSTER_ID>**

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--label **<LABEL>**

Label for the key

--session

Creates a session key that exists only in the current session. The key cannot be recovered after the session ends

--key-length-bytes **<KEY_LENGTH_BYTES>**

Key length in bytes

--attributes [**<KEY_ATTRIBUTES>**...]

Space separated list of key attributes to set for the generated generic secret key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE

--share-crypto-users [**<SHARE_CRYPTO_USERS>**...]

Space separated list of Crypto User usernames to share the generic secret key with

--manage-key-quorum-value **<MANAGE_KEY_QUORUM_VALUE>**

The quorum value for key management operations

--use-key-quorum-value **<USE_KEY_QUORUM_VALUE>**

The quorum value for key usage operations

-h, --help

Print help

Exemplos

Estes exemplos mostram como usar o comando `key generate-symmetric generic-secret` para criar uma chave secreta genérica.

Example Exemplo: criar uma chave secreta genérica

```
aws-cloudhsm > key generate-symmetric generic-secret \  
--label example-generic-secret \  
--key-length-bytes 256  
{  
  "error_code": 0,  
  "data": {  
    "key": {  
      "key-reference": "0x000000000002e08fd",  
      "key-info": {  
        "key-owners": [  
          {  
            "username": "cu1",  
            "key-coverage": "full"  
          }  
        ],  
        "shared-users": [],  
        "key-quorum-values": {  
          "manage-key-quorum-value": 0,  
          "use-key-quorum-value": 0  
        },  
        "cluster-coverage": "full"  
      },  
      "attributes": {  
        "key-type": "generic-secret",  
        "label": "example-generic-secret",  
        "id": "",  
        "class": "secret-key",  
        "encrypt": false,  
        "decrypt": false,  
        "token": true,  
        "always-sensitive": true,  
        "derive": false,  
        "destroyable": true,  
        "extractable": true,  
        "local": true,  
        "modifiable": true,  
        "never-extractable": false,
```

```

    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 256
  }
}
}
}

```

Example Exemplo: criar uma chave secreta genérica com atributos opcionais

```

aws-cloudhsm > key generate-symmetric generic-secret \
--label example-generic-secret \
--key-length-bytes 256 \
--attributes encrypt=true
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000002e08fd",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",

```

```
    "class": "secret-key",
    "encrypt": true,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 256
  }
}
```

Example Exemplo: criar uma chave secreta genérica com valores de quórum

Ao gerar uma chave com controles de quórum, a chave deve estar associada a um número mínimo de usuários igual ao maior valor de quórum da chave. Os usuários associados incluem o proprietário da chave e os usuários de criptografia com quem essa chave é compartilhada. Para definir o número mínimo de usuários com quem compartilhar a chave, obtenha o maior valor de quórum entre o valor do quórum de uso da chave e o valor do quórum de gerenciamento de chaves e subtraia 1 para contabilizar o proprietário da chave, que, por padrão, está associado à chave. Para compartilhar a chave com mais usuários, use o comando [Compartilhar uma chave usando a CloudHSM CLI](#).

```
aws-cloudhsm > key generate-symmetric generic-secret \
--label example-generic-secret \
--key-length-bytes 256 \
--attributes encrypt=true
--share-crypto-users cu2 cu3 cu4 \
--manage-key-quorum-value 4 \
--use-key-quorum-value 2
```

```
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000002e08fd",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu3",
            "key-coverage": "full"
          },
          {
            "username": "cu4",
            "key-coverage": "full"
          }
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 4,
          "use-key-quorum-value": 2
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",
        "class": "secret-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,

```

```
    "local": true,  
    "modifiable": true,  
    "never-extractable": false,  
    "private": true,  
    "sensitive": true,  
    "sign": true,  
    "trusted": false,  
    "unwrap": false,  
    "verify": true,  
    "wrap": false,  
    "wrap-with-trusted": false,  
    "key-length-bytes": 256  
  }  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave AES gerada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo, sign=true).

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

<KEY-LENGTH-BYTES>

Especifica o tamanho da chave em bytes.

Valores válidos:

- 1 a 800

Obrigatório: Sim

<LABEL>

Especifica um rótulo definido pelo usuário para a chave secreta genérica. O tamanho máximo permitido para `label` é de 127 caracteres no Client SDK 5.11 e versões posteriores. O Client SDK 5.10 e versões anteriores têm um limite de 126 caracteres.

Obrigatório: Sim

<SESSION>

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Por padrão, as chaves geradas são chaves persistentes (tokens). Passar `<SESSION>` muda isso, garantindo que uma chave gerada com esse argumento seja uma chave de sessão (efêmera).

Obrigatório: não

<SHARE_CRYPTO_USERS>

Lista separada por espaço de nomes de usuário dos Usuários de criptografia com os quais compartilhar a chave secreta genérica

Obrigatório: não

<MANAGE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de gerenciamento da chave. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

<USE_KEY_QUORUM_VALUE>

O valor do quórum para as operações de uso da chave. Esse valor deve ser menor que ou igual ao número de usuários ao qual essa chave está associada. Isso inclui os usuários com quem a chave é compartilhada e o proprietário da chave. Valor máximo é 8.

Obrigatório: não

Tópicos relacionados

- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

Importar uma chave no formato PEM com a CLI do CloudHSM

Use o `key import pem` comando in AWS CloudHSM para importar uma chave de formato PEM para um módulo de segurança de hardware (HSM). Você pode usá-lo para importar chaves públicas que foram geradas fora do HSM.

Note

Use o comando [Exportar uma chave assimétrica com a CloudHSM CLI](#) para criar um arquivo PEM padrão a partir de uma chave pública ou para criar um arquivo PEM de referência a partir de uma chave privada.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key import pem
```

```
Import key from a PEM file
```

```
Usage: key import pem [OPTIONS] --path <PATH> --label <LABEL> --key-type-  
class <KEY_TYPE_CLASS>
```

```
Options:
```



```

--cluster-id <CLUSTER_ID>
    Unique Id to choose which of the clusters in the config file to run the
    operation against. If not provided, will fall back to the value provided when
    interactive mode was started, or error
--path <PATH>
    Path where the key is located in PEM format
--label <LABEL>
    Label for the imported key
--key-type-class <KEY_TYPE_CLASS>
    Key type and class of the imported key [possible values: ec-public, rsa-
public]
--attributes [<IMPORT_KEY_ATTRIBUTES>...]
    Space separated list of key attributes in the form of
KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the imported key
-h, --help
    Print help

```

Exemplos

Este exemplo mostra como usar o comando `key import pem` para importar uma chave pública RSA de um arquivo no formato PEM.

Example Exemplo: importar uma chave pública RSA

```

aws-cloudhsm > key import pem --path /home/example --label example-imported-key --key-
type-class rsa-public
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001e08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        }
      }
    }
  }
}

```

```

    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "rsa",
    "label": "example-imported-key",
    "id": "0x",
    "check-value": "0x99fe93",
    "class": "public-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": false,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
      "0x8e9c172c37aa22ed1ce25f7c3a7c936dadcd532201400128b044ebb4b96#..3e4930ab910df5a2896eae8853cfe"
    "modulus-size-bits": 2048
  }
},
"message": "Successfully imported key"
}
}

```

Example Exemplo: importar uma chave pública RSA com atributos opcionais

```

aws-cloudhsm > key import pem --path /home/example --label example-imported-key-with-
attributes --key-type-class rsa-public --attributes verify=true
{
  "error_code": 0,

```

```
"data": {
  "key": {
    "key-reference": "0x000000000001e08e3",
    "key-info": {
      "key-owners": [
        {
          "username": "cu1",
          "key-coverage": "full"
        }
      ],
      "shared-users": [],
      "key-quorum-values": {
        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
      },
      "cluster-coverage": "full"
    },
    "attributes": {
      "key-type": "rsa",
      "label": "example-imported-key-with-attributes",
      "id": "0x",
      "check-value": "0x99fe93",
      "class": "public-key",
      "encrypt": false,
      "decrypt": false,
      "token": true,
      "always-sensitive": false,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": false,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": false,
      "sign": false,
      "trusted": false,
      "unwrap": false,
      "verify": true,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 512,
      "public-exponent": "0x010001",
```

```
    "modulus":  
      "0x8e9c172c37aa22ed1ce25f7c3a7c936dadcd532201400128b044ebb4b96#...3e4930ab910df5a2896eae8853cfe"  
      "modulus-size-bits": 2048  
    },  
    "message": "Successfully imported key"  
  }  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PATH>

Especifica o caminho em que o arquivo de chave está localizado.

Obrigatório: Sim

<LABEL>

Especifica o rótulo definido pelo usuário para a chave importada. O tamanho máximo para label é de 126 caracteres.

Obrigatório: Sim

<KEY_TYPE_CLASS>

Tipo e classe da chave encapsulada.

Possíveis valores:

- ec-public
- rsa-public

Obrigatório: Sim

<IMPORT_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços dos atributos de chave a serem definidos para a chave importada na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE (por exemplo,

sign=true). Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: não

Tópicos relacionados

- [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
- [A categoria de verificação de criptografia na CLI do CloudHSM](#)

Listar chaves para um usuário com a CloudHSM CLI

Use o key list comando na CLI do CloudHSM para encontrar todas as chaves do usuário atual presente em seu cluster. AWS CloudHSM A saída inclui chaves que o usuário possui e compartilha, bem como todas as chaves públicas no cluster CloudHSM.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Administradores () COs
- Usuários de criptomoedas (CUs)

Sintaxe

```
aws-cloudhsm > help key list
```

```
List the keys the current user owns, shares, and all public keys in the HSM cluster
```

```
Usage: key list [OPTIONS]
```

```
Options:
```

```
--cluster-id <CLUSTER_ID>
```

```
    Unique Id to choose which of the clusters in the config file to run the  
    operation against. If not provided, will fall back to the value provided when  
    interactive mode was started, or error
```

```
--filter [<FILTER>...]
```

```
    Key reference (e.g. key-reference=0xabc) or space separated list of key  
    attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select  
    matching key(s) to list
```

```
--max-items <MAX_ITEMS>
```

The total number of items to return in the command's output. If the total number of items available is more than the value specified, a next-token is provided in the command's output. To resume pagination, provide the next-token value in the starting-token argument of a subsequent command [default: 10]

`--starting-token <STARTING_TOKEN>`

A token to specify where to start paginating. This is the next-token from a previously truncated response

`-v, --verbose`

If included, prints all attributes and key information for each matched key. By default each matched key only displays its key-reference and label attribute. This flag when used by Admins has no effect

`-h, --help`

Print help

Exemplos

Os exemplos a seguir mostram as diferentes maneiras de executar o comando `key list`. Os exemplos a seguir mostram as saídas como um usuário de criptografia.

Example Exemplo: descobrir todas as chaves – padrão

Esse comando lista as chaves do usuário conectado presente no AWS CloudHSM cluster.

Note

Por padrão, somente 10 chaves do usuário atualmente conectado são exibidas, e somente o `key-reference` e `label` são exibidos como saída. Use as opções de paginação apropriadas para exibir mais ou menos teclas como saída.

```
aws-cloudhsm > key list
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000000003d5",
        "attributes": {
          "label": "test_label_1"
        }
      },
      {
```

```

    "key-reference": "0x00000000000000626",
    "attributes": {
      "label": "test_label_2"
    }
  },.
  ...8 keys later...
],
"total_key_count": 56,
"returned_key_count": 10,
"next_token": "10"
}
}

```

Example Exemplo: encontrar todas as chaves – detalhado

A saída inclui chaves que o usuário possui e compartilha, bem como todas as chaves públicas no HSMs.

Note

Observação: por padrão, somente 10 chaves do usuário atualmente conectado são exibidas. Use as opções de paginação apropriadas para exibir mais ou menos teclas como saída.

```

aws-cloudhsm > key list --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000012000c",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          }
        }
      }
    ]
  }
}

```

```

    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "ec",
    "label": "ec-test-private-key",
    "id": "",
    "check-value": "0x2a737d",
    "class": "private-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 122,
    "ec-point":
"0x0442d53274a6c0ec1a23c165dcb9ccdd72c64e98ae1a9594bb5284e752c746280667e11f1e983493c1c605e0a80
    "curve": "secp224r1"
  }
},
{
  "key-reference": "0x0000000000012000d",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "key-quorum-values": {

```



```

        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
},
"attributes": {
    "key-type": "ec",
    "label": "ec-test-public-key",
    "id": "",
    "check-value": "0x2a737d",
    "class": "public-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 57,
    "ec-point":
"0x0442d53274a6c0ec1a23c165dcb9ccdd72c64e98ae1a9594bb5284e752c746280667e11f1e983493c1c605e0a80
    "curve": "secp224r1"
}
}
],
    ...8 keys later...
    "total_key_count": 1580,
    "returned_key_count": 10
}
}

```

Example Exemplo: retorno paginado

O exemplo a seguir exibe um subconjunto paginado das chaves que mostra somente duas chaves. Em seguida, o exemplo fornece uma chamada subsequente para exibir as próximas duas teclas.

```
aws-cloudhsm > key list --verbose --max-items 2
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000000000030",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "98a6688d1d964ed7b45b9cec5c4b1909",
          "id": "",
          "check-value": "0xb28a46",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
```

```

        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 32
    }
},
{
    "key-reference": "0x00000000000000042",
    "key-info": {
        "key-owners": [
            {
                "username": "cu1",
                "key-coverage": "full"
            }
        ],
        "shared-users": [],
        "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
    },
    "attributes": {
        "key-type": "aes",
        "label": "4ad6cdcbc02044e09fa954143efde233",
        "id": "",
        "check-value": "0xc98104",
        "class": "secret-key",
        "encrypt": true,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,

```

```

        "trusted": false,
        "unwrap": true,
        "verify": true,
        "wrap": true,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
    }
},
"total_key_count": 1580,
"returned_key_count": 2,
"next_token": "2"
}
}

```

Para exibir as próximas 2 teclas, uma chamada subsequente pode ser feita:

```

aws-cloudhsm > key list --verbose --max-items 2 --starting-token 2
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000000081",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "6793b8439d044046982e5b895791e47f",
          "id": "",
          "check-value": "0x3f986f",

```

```

    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 32
  }
},
{
  "key-reference": "0x00000000000000089",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    }
  },
  "cluster-coverage": "full"
},
{
  "attributes": {
    "key-type": "aes",
    "label": "56b30fa05c6741faab8f606d3b7fe105",
    "id": "",
    "check-value": "0xe9201a",
    "class": "secret-key",

```

```

        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 32
    }
},
"total_key_count": 1580,
"returned_key_count": 2,
"next_token": "4"
}
}

```

Para ver mais exemplos que demonstram como o mecanismo de filtragem de chaves funciona na CloudHSM CLI, consulte [Filtrar chaves usando a CLI do CloudHSM](#).

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chaves na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar chaves correspondentes à lista.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte [Atributos de chave da CloudHSM CLI](#)

Obrigatório: não

<MAX_ITEMS>

O número total de itens para retornar na saída do comando. Se o número total de itens disponíveis for maior que o valor especificado, um NextToken é fornecido na saída do comando. Para retomar a paginação, forneça o valor do NextToken no argumento starting-token de um comando subsequente.

Obrigatório: não

<STARTING_TOKEN>

Um token para especificar onde iniciar a paginação. Esse é o NextToken de uma resposta truncada anteriormente.

Obrigatório: não

<VERBOSE>

Se incluído, imprime todos os atributos e as principais informações de cada chave correspondente. Por padrão, cada chave correspondente exibe apenas sua referência de chave e atributo de rótulo. Esse sinalizador, quando usado por administradores, não tem efeito.

Obrigatório: não

Tópicos relacionados

- [Excluir uma chave com a CLI do CloudHSM](#)
- [Exportar uma chave assimétrica com a CloudHSM CLI](#)
- [Cancelar o compartilhamento de uma chave usando a CloudHSM CLI](#)
- [Atributos de chave da CloudHSM CLI](#)
- [Filtrar chaves usando a CLI do CloudHSM](#)

Replicar uma chave com a CLI do CloudHSM

Use o key replicate comando na CLI do CloudHSM para replicar uma chave de um cluster de origem para um cluster de destino AWS CloudHSM . AWS CloudHSM

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Administradores () COs
- Usuários de criptomoedas (CUs)

Note

Os usuários criptográficos devem possuir a chave para usar esse comando.

Requisitos

- Os clusters de origem e destino devem ser clones. Isso significa que um foi criado a partir de um backup do outro, ou ambos foram criados a partir de um backup comum. Consulte [Criar clusters de backups](#) para obter mais informações.
- O proprietário da chave deve existir no cluster de destino. Além disso, se a chave for compartilhada com quaisquer usuários, esses usuários também deverão existir no cluster de destino.
- Para executar esse comando, é necessário estar registrado como usuário de criptografia ou administrador nos clusters de origem e de destino.
- No modo de comando único, o comando usará as variáveis ambientais CLOUDHSM_PIN e CLOUDHSM_ROLE para se autenticar no cluster de origem. Consulte [Modo de comando único](#) para obter mais informações. Para fornecer credenciais ao cluster de destino, é preciso definir duas variáveis ambientais adicionais: DESTINATION_CLOUDHSM_PIN e DESTINATION_CLOUDHSM_ROLE:

```
$ export DESTINATION_CLOUDHSM_ROLE=<role>
```

```
$ export DESTINATION_CLOUDHSM_PIN=<username:password>
```

- No modo interativo, os usuários precisarão fazer login explícito nos clusters de origem e de destino.

Sintaxe

```
aws-cloudhsm > help key replicate
```

Replicate a key from a source to a destination cluster

Usage: key replicate --filter [<FILTER>...] --source-cluster-id <SOURCE_CLUSTER_ID> --destination-cluster-id <DESTINATION_CLUSTER_ID>

Options:

```
--filter [<FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select matching key on the source cluster

```
--source-cluster-id <SOURCE_CLUSTER_ID>
```

Source cluster ID

```
--destination-cluster-id <DESTINATION_CLUSTER_ID>
```

Destination cluster ID

```
-h, --help
```

Print help

Exemplos

Example Exemplo: replicar chave

Esse comando replica uma chave de um cluster de origem para um cluster de destino clonado. O exemplo abaixo demonstra a saída quando conectado como usuário criptográfico em ambos os clusters.

```
crypto-user-1@cluster-1234abcdefg > key replicate \  
  --filter attr.label=example-key \  
  --source-cluster-id cluster-1234abcdefg \  
  --destination-cluster-id cluster-2345bcdefgh
```

```
{  
  "error_code": 0,  
  "data": {  
    "key": {  
      "key-reference": "0x00000000000300006",  
      "key-info": {  
        "key-owners": [  
          {  
            "username": "crypto-user-1",  
            "key-coverage": "full"  
          }  
        ]  
      }  
    }  
  }  
}
```

```
    ],
    "shared-users": [],
    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "aes",
    "label": "example-key",
    "id": "0x",
    "check-value": "0x5e118e",
    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": true,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
},
"message": "Successfully replicated key"
}
```

Argumentos

<FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave correspondente no cluster de origem.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte [Atributos de chave da CloudHSM CLI](#)

Obrigatório: Sim

<SOURCE_CLUSTER_ID>

O ID do cluster de origem.

Obrigatório: Sim

<DESTINATION_CLUSTER_ID>

O ID do cluster de destino.

Obrigatório: Sim

Tópicos relacionados

- [Conectar-se a vários clusters com a CLI do CloudHSM](#)

Definir os atributos das chaves com a CloudHSM CLI

Use o `key set-attribute` comando na CLI do CloudHSM para definir os atributos das chaves no seu cluster. AWS CloudHSM Somente o CU que criou a chave e, conseqüentemente, a possui pode alterar os atributos da chave.

Para obter uma lista dos principais atributos que podem ser usados na CloudHSM CLI, consulte [Atributos de chave da CloudHSM CLI](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários criptográficos (CUs) podem executar esse comando.

- Os administradores podem definir o atributo confiável.

Requisitos

Para executar esse comando, você deve estar registrado como um CU. Para definir o atributo confiável, você deve estar logado como um usuário administrador.

Sintaxe

```
aws-cloudhsm > help key set-attribute
```

Set an attribute for a key in the HSM cluster

Usage: cloudhsm-cli key set-attribute [OPTIONS] --filter [**<FILTER>**...] --name **<KEY_ATTRIBUTE>** --value **<KEY_ATTRIBUTE_VALUE>**

Options:

--cluster-id <CLUSTER_ID>	Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error
--filter [<FILTER> ...]	Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key to modify
--name <KEY_ATTRIBUTE>	Name of attribute to be set
--value <KEY_ATTRIBUTE_VALUE> ...	Attribute value to be set
--approval <APPROVAL>	Filepath of signed quorum token file to approve operation
-h, --help	Print help

Exemplo: configuração de um atributo de chave

O exemplo a seguir mostra como usar o comando key set-attribute para definir o rótulo.

Example

1. Use a chave com o rótulo my_key, conforme mostrado aqui:

```
aws-cloudhsm > key set-attribute --filter attr.label=my_key --name encrypt --value false
{
  "error_code": 0,
  "data": {
    "message": "Attribute set successfully"
```

```
}
}
```

2. Use o comando `key list` para confirmar que o atributo `encrypt` foi alterado:

```
aws-cloudhsm > key list --filter attr.label=my_key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000006400ec",
        "key-info": {
          "key-owners": [
            {
              "username": "bob",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "my_key",
          "id": "",
          "check-value": "0x6bd9f7",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": true,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
```

```
        "sign": true,
        "trusted": true,
        "unwrap": true,
        "verify": true,
        "wrap": true,
        "wrap-with-trusted": false,
        "key-length-bytes": 32
    }
}
],
"total_key_count": 1,
"returned_key_count": 1
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<KEY_ATTRIBUTE>

Especifica o nome do atributo de chave.

Obrigatório: Sim

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis com a CloudHSM CLI, consulte [Atributos de chave da CloudHSM CLI](#)

Obrigatório: não

<KEY_ATTRIBUTE_VALUE>

A chave especifica o nome do atributo.

Obrigatório: Sim

<KEY_REFERENCE>

Uma representação hexadecimal ou decimal da chave. (como uma alça de chave).

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave for maior do que 1.

Tópicos relacionados

- [Filtrar chaves usando a CLI do CloudHSM](#)
- [Atributos de chave da CloudHSM CLI](#)

Compartilhar uma chave usando a CloudHSM CLI

Use o key share comando na CLI do CloudHSM para compartilhar uma chave CUs com outras pessoas em seu cluster. AWS CloudHSM

Somente a UC que criou a chave e, conseqüentemente, a possui pode cancelar o compartilhamento da chave. Os usuários com quem a chave é compartilhada podem usá-la em operações criptográficas, mas não podem excluí-la, exportá-la, compartilhá-la nem descompartilhá-la. Além disso, esses usuários não podem alterar [os principais atributos](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key share  
Share a key in the HSM cluster with another user
```

Usage: key share --filter [<FILTER>...] --username <USERNAME> --role <ROLE>

Options:

--cluster-id <CLUSTER_ID>

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--filter [<FILTER>...]

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key for sharing

--username <USERNAME>

A username with which the key will be shared

--role <ROLE>

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

--approval <APPROVAL>

Filepath of signed quorum token file to approve operation

-h, --help

Print help (see a summary with '-h')

Exemplo: compartilhar uma chave com outro CU

O exemplo a seguir mostra como usar o comando key share para compartilhar uma chave com a CU alice.

Example

1. Execute o comando key share com o qual compartilhar a chave alice.

```
aws-cloudhsm > key share --filter attr.label="rsa_key_to_share" attr.class=private-
key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
```



```
    "message": "Key shared successfully"
  }
}
```

2. Execute o comando key list.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" attr.class=private-
key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
```

```

        "key-coverage": "full"
    },
    {
        "username": "alice",
        "key-coverage": "full"
    }
],
"key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
},
"cluster-coverage": "full"
},
"attributes": {
    "key-type": "rsa",
    "label": "rsa_key_to_share",
    "id": "",
    "check-value": "0xae8ff0",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1219,
    "public-exponent": "0x010001",
    "modulus":
"0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254",
    "modulus-size-bits": 2048
}
}

```

```
    ],  
    "total_key_count": 1,  
    "returned_key_count": 1  
  }  
}
```

3. Na lista acima, verifique se alice está na lista de shared-users

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: Sim

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_). O nome de usuário não diferencia maiúsculas de minúsculas neste comando. O nome de usuário é sempre exibido em minúsculas.

Obrigatório: Sim

<ROLE>

Especifica a função atribuída a esse usuário. Esse parâmetro é obrigatório. Para obter a função do usuário, use o comando lista de usuário. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Tipos de usuários do HSM para a CLI do CloudHSM](#).

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave for maior do que 1.

Tópicos relacionados

- [Filtrar chaves usando a CLI do CloudHSM](#)
- [Atributos de chave da CloudHSM CLI](#)

Cancelar o compartilhamento de uma chave usando a CloudHSM CLI

Use o `key unshare` comando na CLI do CloudHSM para cancelar o compartilhamento de uma chave com outras pessoas em seu cluster. CUs AWS CloudHSM

Somente o CU que criou a chave e, conseqüentemente, a possui pode cancelar o compartilhamento da chave. Os usuários com quem a chave é compartilhada podem usá-la em operações criptográficas, mas não podem excluí-la, exportá-la, compartilhá-la nem descompartilhá-la. Além disso, esses usuários não podem alterar [os principais atributos](#).

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unshare
Unshare a key in the HSM cluster with another user

Usage: key unshare --filter [<FILTER>...] --username <USERNAME> --role <ROLE>

Options:
    --cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--filter [<FILTER>...]`

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key for unsharing

`--username <USERNAME>`

A username with which the key will be unshared

`--role <ROLE>`

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

`--approval <APPROVAL>`

Filepath of signed quorum token file to approve operation

`-h, --help`

Print help (see a summary with '-h')

Exemplo: cancelar o compartilhamento de uma chave com outro CU

O exemplo a seguir mostra como usar o comando `key unshare` para descompartilhar uma chave com a CU `alice`.

Example

1. Execute o comando `key list` e filtre pela chave específica com a qual você deseja cancelar o compartilhamento com `alice`.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" attr.class=private-key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000001c0686",
```

```
"key-info": {
  "key-owners": [
    {
      "username": "cu3",
      "key-coverage": "full"
    }
  ],
  "shared-users": [
    {
      "username": "cu2",
      "key-coverage": "full"
    },
    {
      "username": "cu1",
      "key-coverage": "full"
    },
    {
      "username": "cu4",
      "key-coverage": "full"
    },
    {
      "username": "cu5",
      "key-coverage": "full"
    },
    {
      "username": "cu6",
      "key-coverage": "full"
    },
    {
      "username": "cu7",
      "key-coverage": "full"
    },
    {
      "username": "alice",
      "key-coverage": "full"
    }
  ],
  "key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
  },
  "cluster-coverage": "full"
},
"attributes": {
```

```

    "key-type": "rsa",
    "label": "rsa_key_to_share",
    "id": "",
    "check-value": "0xae8ff0",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": true,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1219,
    "public-exponent": "0x010001",
    "modulus":
    "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
    "modulus-size-bits": 2048
  }
],
"total_key_count": 1,
"returned_key_count": 1
}
}

```

2. Confirme que alice está na saída `shared-users` e execute o comando `key unshare` a seguir para cancelar o compartilhamento da chave com alice.

```

aws-cloudhsm > key unshare --filter attr.label="rsa_key_to_share"
attr.class=private-key --username alice --role crypto-user
{
  "error_code": 0,

```

```
"data": {  
  "message": "Key unshared successfully"  
}  
}
```

3. Execute o comando `key list` novamente para confirmar que a chave não foi compartilhada com `alice`.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" attr.class=private-key --verbose  
{  
  "error_code": 0,  
  "data": {  
    "matched_keys": [  
      {  
        "key-reference": "0x000000000001c0686",  
        "key-info": {  
          "key-owners": [  
            {  
              "username": "cu3",  
              "key-coverage": "full"  
            }  
          ],  
          "shared-users": [  
            {  
              "username": "cu2",  
              "key-coverage": "full"  
            },  
            {  
              "username": "cu1",  
              "key-coverage": "full"  
            },  
            {  
              "username": "cu4",  
              "key-coverage": "full"  
            },  
            {  
              "username": "cu5",  
              "key-coverage": "full"  
            },  
            {  
              "username": "cu6",  
              "key-coverage": "full"  
            }  
          ]  
        }  
      }  
    ]  
  }  
}
```



```

    },
    {
      "username": "cu7",
      "key-coverage": "full"
    },
  ],
  "key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
  },
  "cluster-coverage": "full"
},
"attributes": {
  "key-type": "rsa",
  "label": "rsa_key_to_share",
  "id": "",
  "check-value": "0xae8ff0",
  "class": "private-key",
  "encrypt": false,
  "decrypt": true,
  "token": true,
  "always-sensitive": true,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": true,
  "modifiable": true,
  "never-extractable": false,
  "private": true,
  "sensitive": true,
  "sign": true,
  "trusted": false,
  "unwrap": true,
  "verify": false,
  "wrap": false,
  "wrap-with-trusted": false,
  "key-length-bytes": 1219,
  "public-exponent": "0x010001",
  "modulus":
    "0xa8855c9a933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254",
  "modulus-size-bits": 2048
}
}
],

```

```
"total_key_count": 1,  
"returned_key_count": 1  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis, consulte [Atributos de chave da CloudHSM CLI](#).

Obrigatório: Sim

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (`_`). O nome de usuário não diferencia maiúsculas de minúsculas neste comando. O nome de usuário é sempre exibido em minúsculas.

Obrigatório: Sim

<ROLE>

Especifica a função atribuída a esse usuário. Esse parâmetro é obrigatório. Para obter a função do usuário, use o comando `lista de usuário`. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Tipos de usuários do HSM para a CLI do CloudHSM](#).

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave for maior do que 1.

Tópicos relacionados

- [Filtrar chaves usando a CLI do CloudHSM](#)
- [Atributos de chave da CloudHSM CLI](#)

O comando key unwrap na CLI do CloudHSM

O comando principal key unwrap na CLI do CloudHSM importa uma chave privada simétrica ou assimétrica criptografada (encapsulada) de um arquivo para o Hardware Security Module (HSM – Módulo de segurança de hardware). Esse comando é projetado para importar chaves criptografadas que foram encapsuladas pelo comando [O comando key wrap na CLI do CloudHSM](#), mas também pode ser usado para desencapsular chaves que foram encapsuladas com outras ferramentas. No entanto, nessas situações, recomendamos usar as bibliotecas de software PKCS#11 ou JCE para desencapsular a chave.

- [aes-gcm](#)
- [aes-no-pad](#)
- [aes-pkcs5-pad](#)
- [aes-zero-pad](#)
- [cloudhsm-aes-gcm](#)
- [rsa-aes](#)
- [rsa-oaep](#)
- [rsa-pkcs](#)

Desencapsular uma chave com o AES-GCM usando a CloudHSM CLI

Use o comando key unwrap aes-gcm na CloudHSM CLI para desencapsular uma chave de carga útil no cluster usando a chave de encapsulamento AES e o mecanismo de desencapsulamento AES-GCM.

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o `key unwrap aes-gcm` comando, você deve ter a chave de encapsulamento AES em seu AWS CloudHSM cluster e seu `unwrap` atributo deve estar definido como `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap aes-gcm
```

```
Usage: key unwrap aes-gcm [OPTIONS] --filter [<FILTER>...] --tag-length-
bits <TAG_LENGTH_BITS> --key-type-class <KEY_TYPE_CLASS> --label <LABEL> --iv <IV> <--
data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a key to unwrap with

```
--data-path <DATA_PATH>
```

Path to the binary file containing the wrapped key data

```
--data <DATA>
```

Base64 encoded wrapped key data

```
--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
```

Space separated list of key attributes in the form of `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` for the unwrapped key

```
--share-crypto-users [<SHARE_CRYPTO_USERS;...>]
```

Space separated list of Crypto User usernames to share the unwrapped key with

```

--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>
    The quorum value for key management operations for the unwrapped key
--use-key-quorum-value <USE_KEY_QUORUM_VALUE>
    The quorum value for key usage operations for the unwrapped key
--aad <AAD>
    Aes GCM Additional Authenticated Data (AAD) value, in hex
--tag-length-bits <TAG_LENGTH_BITS>
    Aes GCM tag length in bits
--key-type-class <KEY_TYPE_CLASS>
    Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
--label <LABEL>
    Label for the unwrapped key
--session
    Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
--iv <IV>
    Initial value used to wrap the key, in hex
--approval <APPROVAL>
    Filepath of signed quorum token file to approve operation
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar o comando `key unwrap aes-gcm` usando uma chave AES com o valor do atributo `unwrap` definido como `true`.

Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```

aws-cloudhsm > key unwrap aes-gcm --key-type-class aes --label aes-unwrapped
--filter attr.label=aes-example --tag-length-bits 64 --aad 0x10 --iv
0xf90613bb8e337ec0339aad21 --data xvslgrtg8kHrzvekn97tLSieokpPwV8
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x0000000000001808e4",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",

```

```
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "key-quorum-values": {
      "manage-key-quorum-value": 0,
      "use-key-quorum-value": 0
    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "aes",
    "label": "aes-unwrapped",
    "id": "0x",
    "check-value": "0x8d9099",
    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": false,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
}
}
```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```
aws-cloudhsm > key unwrap aes-gcm --key-type-class aes --label aes-unwrapped
--filter attr.label=aes-example --tag-length-bits 64 --aad 0x10 --iv
0xf90613bb8e337ec0339aad21 --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001808e4",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
```

```
    "sign": true,  
    "trusted": false,  
    "unwrap": false,  
    "verify": true,  
    "wrap": false,  
    "wrap-with-trusted": false,  
    "key-length-bytes": 16  
  }  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços contendo atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para a chave encapsulada.

Obrigatório: não

<AAD>

Valor de dados autenticados adicionais (AAD) do Aes GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Comprimento de tag do Aes GCM em bits.

Obrigatório: Sim

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: aes, des3, ec-private, generic-secret, rsa-private].

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<IV>

Valor inicial usado para encapsular a chave, em hexadecimal.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desembrulhe uma chave AES-NO-PAD usando a CLI do CloudHSM

Use o key unwrap aes-no-pad comando na CLI do CloudHSM para desempacotar uma chave de carga no cluster usando a chave de empacotamento AES e o AWS CloudHSM mecanismo de desempacotamento. AES-NO-PAD

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o key unwrap aes-no-pad comando, você deve ter a chave de encapsulamento AES em seu AWS CloudHSM cluster e seu unwrap atributo deve estar definido como. `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap aes-no-pad
```

```
Usage: key unwrap aes-no-pad [OPTIONS] --filter [<FILTER>...] --key-type-  
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key to unwrap with

- `--data-path <DATA_PATH>`
Path to the binary file containing the wrapped key data
- `--data <DATA>`
Base64 encoded wrapped key data
- `--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]`
Space separated list of key attributes in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
- `--share-crypto-users [<SHARE_CRYPTO_USERS>...]`
Space separated list of Crypto User usernames to share the unwrapped key with
- `--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>`
The quorum value for key management operations for the unwrapped key
- `--use-key-quorum-value <USE_KEY_QUORUM_VALUE>`
The quorum value for key usage operations for the unwrapped key
- `--key-type-class <KEY_TYPE_CLASS>`
Key type and class of wrapped key [possible values: aes, des3, ec-private, generic-secret, rsa-private]
- `--label <LABEL>`
Label for the unwrapped key
- `--session`
Creates a session key that exists only in the current session. The key cannot be recovered after the session ends
- `--approval <APPROVAL>`
Filepath of signed quorum token file to approve operation
- `-h, --help`
Print help

Exemplos

Esses exemplos mostram como usar o comando `key unwrap aes-no-pad` usando uma chave AES com o valor do atributo `unwrap` definido como `true`.

Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```
aws-cloudhsm > key unwrap aes-no-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data eXK3PMA0nKM9y3YX6brbhtMoC060E0H9
{
  "error_code": 0,
  "data": {
    "key": {
```

```

    "key-reference": "0x000000000001c08ec",
    "key-info": {
      "key-owners": [
        {
          "username": "cu1",
          "key-coverage": "full"
        }
      ],
      "shared-users": [],
      "key-quorum-values": {
        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
      },
      "cluster-coverage": "full"
    },
    "attributes": {
      "key-type": "aes",
      "label": "aes-unwrapped",
      "id": "0x",
      "check-value": "0x8d9099",
      "class": "secret-key",
      "encrypt": false,
      "decrypt": false,
      "token": true,
      "always-sensitive": false,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": false,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": true,
      "trusted": false,
      "unwrap": false,
      "verify": true,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 16
    }
  }
}

```

```
}
```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```
aws-cloudhsm > key unwrap aes-no-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08ec",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
```

```
    "sensitive": true,  
    "sign": true,  
    "trusted": false,  
    "unwrap": false,  
    "verify": true,  
    "wrap": false,  
    "wrap-with-trusted": false,  
    "key-length-bytes": 16  
  }  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços contendo atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para a chave encapsulada.

Obrigatório: não

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: aes, des3, ec-private, generic-secret, rsa-private].

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desempacote uma chave com o AES- PKCS5 -PAD usando a CLI do CloudHSM

Use o comando key unwrap aes-pkcs5-pad na CloudHSM CLI para desencapsular uma chave de carga útil usando a chave de encapsulamento AES e o mecanismo de desencapsulamento AES- PKCS5 -PAD.

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o key unwrap aes-pkcs5-pad comando, você deve ter a chave de encapsulamento AES em seu AWS CloudHSM cluster e seu unwrap atributo deve estar definido como. true

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap aes-pkcs5-pad
```

```
Usage: key unwrap aes-pkcs5-pad [OPTIONS] --filter [<FILTER>...] --key-type-  
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key to unwrap with

```
--data-path <DATA_PATH>
```

Path to the binary file containing the wrapped key data

```
--data <DATA>
```

Base64 encoded wrapped key data

```
--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
```

Space separated list of key attributes in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key

```
--share-crypto-users [<SHARE_CRYPTO_USERS>...]
```

Space separated list of Crypto User usernames to share the unwrapped key with

```
--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>;
```

The quorum value for key management operations for the unwrapped key

```
--use-key-quorum-value <USE_KEY_QUORUM_VALUE>;
```

The quorum value for key usage operations for the unwrapped key

```
--key-type-class <KEY_TYPE_CLASS>
```



```

    Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
    --label <LABEL>
        Label for the unwrapped key
    --session
        Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
    --approval <APPROVAL>
        Filepath of signed quorum token file to approve operation
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar o comando `key unwrap aes-pkcs5-pad` usando uma chave AES com o valor do atributo `unwrap` definido como `true`.

Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```

aws-cloudhsm > key unwrap aes-pkcs5-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data MbuYNresf0KyGNnxKWen88nSfX+uUE/0qmGofSisicY=
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",

```

```

    "label": "aes-unwrapped",
    "id": "0x",
    "check-value": "0x8d9099",
    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": false,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
}
}
}

```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```

aws-cloudhsm > key unwrap aes-pkcs5-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ]
      }
    }
  }
}

```

```

    }
  ],
  "shared-users": [],
  "key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
  },
  "cluster-coverage": "full"
},
"attributes": {
  "key-type": "aes",
  "label": "aes-unwrapped",
  "id": "0x",
  "check-value": "0x8d9099",
  "class": "secret-key",
  "encrypt": false,
  "decrypt": false,
  "token": true,
  "always-sensitive": false,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": false,
  "modifiable": true,
  "never-extractable": false,
  "private": true,
  "sensitive": true,
  "sign": true,
  "trusted": false,
  "unwrap": false,
  "verify": true,
  "wrap": false,
  "wrap-with-trusted": false,
  "key-length-bytes": 16
}
}
}
}

```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços contendo atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para a chave encapsulada.

Obrigatório: não

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: `aes`, `des3`, `ec-private`, `generic-secret`, `rsa-private`].

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desembrolhe uma chave AES-ZERO-PAD usando a CLI do CloudHSM

Use o key unwrap aes-zero-pad comando na CLI do CloudHSM para desempacotar uma chave de carga no cluster usando a chave de empacotamento AES e o AWS CloudHSM mecanismo de desempacotamento. AES-ZERO-PAD

As chaves desembrolhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o key unwrap aes-no-pad comando, você deve ter a chave de encapsulamento AES em seu AWS CloudHSM cluster e seu unwrap atributo deve estar definido como. `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap aes-zero-pad
```

```
Usage: key unwrap aes-zero-pad [OPTIONS] --filter [<FILTER>...] --key-type-
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key to unwrap with

```
--data-path <DATA_PATH>
```

Path to the binary file containing the wrapped key data

```
--data <DATA>
```

Base64 encoded wrapped key data

```
--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
```

Space separated list of key attributes in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key

```
--share-crypto-users [<SHARE_CRYPTO_USERS>...]
```

Space separated list of Crypto User usernames to share the unwrapped key with

```
--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>;
```

The quorum value for key management operations for the unwrapped key

```
--use-key-quorum-value <USE_KEY_QUORUM_VALUE>;
```

The quorum value for key usage operations for the unwrapped key

```
--key-type-class <KEY_TYPE_CLASS>
```

Key type and class of wrapped key [possible values: aes, des3, ec-private, generic-secret, rsa-private]

```
--label <LABEL>
```

Label for the unwrapped key

```
--session
```

Creates a session key that exists only in the current session. The key cannot be recovered after the session ends

```
--approval <APPROVAL>
```

Filepath of signed quorum token file to approve operation

```
-h, --help
```

Print help

Exemplos

Esses exemplos mostram como usar o comando key unwrap aes-zero-pad usando uma chave AES com o valor do atributo unwrap definido como true.

Exemplo Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```
aws-cloudhsm > key unwrap aes-zero-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data L1wV1L/YeBNVAw6Mpk3owFJZXBzDL0nt
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08e7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
```

```

    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
}
}
}

```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```

aws-cloudhsm > key unwrap aes-zero-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08e7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,

```



```
    "token": true,  
    "always-sensitive": false,  
    "derive": false,  
    "destroyable": true,  
    "extractable": true,  
    "local": false,  
    "modifiable": true,  
    "never-extractable": false,  
    "private": true,  
    "sensitive": true,  
    "sign": true,  
    "trusted": false,  
    "unwrap": false,  
    "verify": true,  
    "wrap": false,  
    "wrap-with-trusted": false,  
    "key-length-bytes": 16  
  }  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços contendo atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para a chave encapsulada.

Obrigatório: não

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: aes, des3, ec-private, generic-secret, rsa-private].

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desembrulhe uma chave CLOUDHSM-AES-GCM usando a CLI do CloudHSM

Use o `key unwrap cloudhsm-aes-gcm` comando na CLI do CloudHSM para desempacotar uma chave de carga no cluster usando a chave de empacotamento AES e o AWS CloudHSM mecanismo de desempacotamento. CLOUDHSM-AES-GCM

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o `key unwrap cloudhsm-aes-gcm` comando, você deve ter a chave de encapsulamento AES em seu AWS CloudHSM cluster e seu `unwrap` atributo deve estar definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap cloudhsm-aes-gcm
```

```
Usage: key unwrap cloudhsm-aes-gcm [OPTIONS] --filter [<FILTER>...] --tag-length-bits <TAG_LENGTH_BITS> --key-type-class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a key to unwrap with

```
--data-path <DATA_PATH>
```

Path to the binary file containing the wrapped key data

```
--data <DATA>
```

```

    Base64 encoded wrapped key data
--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
    Space separated list of key attributes in the form of
KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
--share-crypto-users [<SHARE_CRYPTO_USERS;...>]
    Space separated list of Crypto User usernames to share the unwrapped key with
--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
    The quorum value for key management operations for the unwrapped key
--use-key-quorum-value <USE_KEY_QUORUM_VALUE;
    The quorum value for key usage operations for the unwrapped key
--aad <AAD>
    Aes GCM Additional Authenticated Data (AAD) value, in hex
--tag-length-bits <TAG_LENGTH_BITS>
    Aes GCM tag length in bits
--key-type-class <KEY_TYPE_CLASS>
    Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
--label <LABEL>
    Label for the unwrapped key
--session
    Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
--approval <APPROVAL>
    Filepath of signed quorum token file to approve operation
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar o comando `key unwrap cloudhsm-aes-gcm` usando uma chave AES com o valor do atributo `unwrap` definido como `true`.

Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```

aws-cloudhsm > key unwrap cloudhsm-aes-gcm --key-type-class aes --label aes-
unwrapped --filter attr.label=aes-example --tag-length-bits 64 --aad 0x10 --data
6Rn8nkjEriDYlnP3P8nPkYQ8hp10EJ899zsrF+aTB0i/f1lZ
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001408e8",

```

```

    "key-info": {
      "key-owners": [
        {
          "username": "cu1",
          "key-coverage": "full"
        }
      ],
      "shared-users": [],
      "key-quorum-values": {
        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
      },
      "cluster-coverage": "full"
    },
    "attributes": {
      "key-type": "aes",
      "label": "aes-unwrapped",
      "id": "0x",
      "check-value": "0x8d9099",
      "class": "secret-key",
      "encrypt": false,
      "decrypt": false,
      "token": true,
      "always-sensitive": false,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": false,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": true,
      "trusted": false,
      "unwrap": false,
      "verify": true,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 16
    }
  }
}

```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```
aws-cloudhsm > key unwrap cloudhsm-aes-gcm --key-type-class aes --label aes-unwrapped
--filter attr.label=aes-example --tag-length-bits 64 --aad 0x10 --data-path payload-
key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001408e8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
```

```
    "sign": true,  
    "trusted": false,  
    "unwrap": false,  
    "verify": true,  
    "wrap": false,  
    "wrap-with-trusted": false,  
    "key-length-bytes": 16  
  }  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços contendo atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para a chave encapsulada.

Obrigatório: não

<AAD>

Valor de dados autenticados adicionais (AAD) do Aes GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Comprimento de tag do Aes GCM em bits.

Obrigatório: Sim

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: aes, des3, ec-private, generic-secret, rsa-private].

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desencapsular uma chave com o RSA-AES usando a CloudHSM CLI

Use o comando `key unwrap rsa-aes` na CloudHSM CLI para desencapsular uma chave de carga útil usando uma chave privada RSA e o mecanismo de desencapsulamento RSA-AES.

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar `key unwrap rsa-aes`, você deve ter a chave privada RSA da chave de empacotamento pública RSA em seu AWS CloudHSM cluster e seu `unwrap` atributo deve estar definido como `true`.

Note

Esse comando está disponível somente com o CloudHSM CLI 5.11+.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap rsa-aes  
Usage: key unwrap rsa-aes [OPTIONS] --filter [<FILTER>...] --hash-  
function <HASH_FUNCTION> --mgf <MGF> --key-type-class <KEY_TYPE_CLASS> --label <LABEL>  
  <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
  --cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
  --filter [<FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key to unwrap with

- `--data-path <DATA_PATH>`
Path to the binary file containing the wrapped key data
- `--data <DATA>`
Base64 encoded wrapped key data
- `--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]`
Space separated list of key attributes in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
- `--share-crypto-users [<SHARE_CRYPTO_USERS>...]`
Space separated list of Crypto User usernames to share the unwrapped key with
- `--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>`
The quorum value for key management operations for the unwrapped key
- `--use-key-quorum-value <USE_KEY_QUORUM_VALUE>`
The quorum value for key usage operations for the unwrapped key
- `--hash-function <HASH_FUNCTION>`
Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
- `--mgf <MGF>`
Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224, mgf1-sha256, mgf1-sha384, mgf1-sha512]
- `--key-type-class <KEY_TYPE_CLASS>`
Key type and class of wrapped key [possible values: aes, des3, ec-private, generic-secret, rsa-private]
- `--label <LABEL>`
Label for the unwrapped key
- `--session`
Creates a session key that exists only in the current session. The key cannot be recovered after the session ends
- `--approval <APPROVAL>`
Filepath of signed quorum token file to approve operation
- `-h, --help`
Print help

Exemplo

Esses exemplos mostram como usar o comando `key unwrap rsa-aes` usando a chave privada RSA com o valor do atributo `unwrap` definido como `true`.

Exemplo Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```
aws-cloudhsm > key unwrap rsa-aes --key-type-class aes --label aes-unwrapped
--filter attr.label=rsa-private-key-example --hash-function sha256 --
mgf mgf1-sha256 --data HrSE1DEyLjIeyGdPa9R+ebiqB5TIJGyamPker31ZebPwRA
+NcerbAJ08DJ1lXPygZcI21vIFSZJuWMEiWpe1R9D/5WSYgxLVKex30xCFqebtEzxbKuv4D0mU4meSofqREYvtb3EoIKwjy
+RL5WGXKe4nAboAkC5G07veI5yHL1SaKlssSJtTL/CFpbSLsAFuYbv/NUCWwMY5mwyVTCSlw+HlgKK
+5TH1MzBaSi8fpfyepLT8sHy2Q/VRl6ifb49p6m0KQFbRVvz/OWUd6l4d97BdgtaEz6ueg==
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x0000000000001808e2",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
```

```

    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
}
}
}

```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```

aws-cloudhsm > key unwrap rsa-aes --key-type-class aes --label aes-unwrapped --filter
attr.label=rsa-private-key-example --hash-function sha256 --mgf mgf1-sha256 --data-
path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x0000000000001808e2",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",

```

```
    "id": "0x",
    "check-value": "0x8d9099",
    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": false,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços contendo atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para a chave encapsulada.

Obrigatório: não

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: aes, des3, ec-private, generic-secret, rsa-private].

Obrigatório: Sim

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<MGF>

Especifica a função de geração da máscara.

 Note

A função hash da função de geração de máscara deve corresponder à função hash do mecanismo de assinatura.

Valores válidos:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desencapsular uma chave com o RSA-OAEP usando a CloudHSM CLI

Use o comando `key unwrap rsa-oaep` na CloudHSM CLI para desencapsular uma chave de carga útil usando a chave privada RSA e o mecanismo de desencapsulamento RSA-OAEP.

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o `key unwrap rsa-oaep` comando, você deve ter a chave privada RSA da chave de empacotamento pública RSA em seu AWS CloudHSM cluster e seu `unwrap` atributo deve estar definido como `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap rsa-oaep
```

```
Usage: key unwrap rsa-oaep [OPTIONS] --filter [<FILTER>...] --hash-  
function <HASH_FUNCTION> --mgf <MGF> --key-type-class <KEY_TYPE_CLASS> --label <LABEL>  
--data-path <DATA_PATH>|--data <DATA>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a key to unwrap with

```
--data-path <DATA_PATH>
```

Path to the binary file containing the wrapped key data

```
--data <DATA>
```



```

    Base64 encoded wrapped key data
--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
    Space separated list of key attributes in the form of
KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
--share-crypto-users [<SHARE_CRYPTO_USERS>...]
    Space separated list of Crypto User usernames to share the unwrapped key with
--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>
    The quorum value for key management operations for the unwrapped key
--use-key-quorum-value <USE_KEY_QUORUM_VALUE>
    The quorum value for key usage operations for the unwrapped key
--hash-function <HASH_FUNCTION>
    Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
--mgf <MGF>
    Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224,
mgf1-sha256, mgf1-sha384, mgf1-sha512]
--key-type-class <KEY_TYPE_CLASS>
    Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
--label <LABEL>
    Label for the unwrapped key
--session
    Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
--approval <APPROVAL>
    Filepath of signed quorum token file to approve operation
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar o comando `key unwrap rsa-oaep` usando a chave privada RSA com o valor do atributo `unwrap` definido como `true`.

Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```

aws-cloudhsm > key unwrap rsa-oaep --key-type-class aes --label aes-unwrapped --filter
attr.label=rsa-private-example-key --hash-function sha256 --mgf mgf1-sha256 --data
OjJe4msobPLz9TuSAdULEu17T5rMDWtS1LyBSkLbaZnYzzpdrhsbGLbwZJCtB/jGkDnDB4qyTA0QwEpggGf6v
+Yx6JcesNeKKNU8XZa1/YBoHC8noTGUSD12qr+u2tDc84NPv6d+F2K00NXsSxMhmxzzNG/
gzTVIJh0uy/B1yHjGP4m0XoDZf5+7f5M1CjxBmz4Vva/wrWHGCSG0y0aWb1Ev0iHAIt3UBdyKmU+//
My4xjfJv7WGGu3DFUUIZ06TihRtKQhUYU1M9u6NPf9riJJfHsk6QCUSZ9yWThDT9as6i7e3htnyDhIhGwaoK8JU855cN/
YNKAUqkNpC4FPL3iw==

```

```
{
  "data": {
    "key": {
      "key-reference": "0x0000000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

```

    }
  }
}

```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```

aws-cloudhsm > key unwrap rsa-oaep --key-type-class aes --label aes-unwrapped --filter
attr.label=rsa-private-example-key --hash-function sha256 --mgf mgf1-sha256 --data-
path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,

```

```
    "modifiable": true,  
    "never-extractable": false,  
    "private": true,  
    "sensitive": true,  
    "sign": true,  
    "trusted": false,  
    "unwrap": false,  
    "verify": true,  
    "wrap": false,  
    "wrap-with-trusted": false,  
    "key-length-bytes": 16  
  }  
}  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços contendo atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE para a chave encapsulada.

Obrigatório: não

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: aes, des3, ec-private, generic-secret, rsa-private].

Obrigatório: Sim

<HASH_FUNCTION>

Especifica a função hash.

Valores válidos:

- sha1
- sha224
- sha256
- sha384
- sha512

Obrigatório: Sim

<MGF>

Especifica a função de geração da máscara.

 Note

A função hash da função de geração de máscara deve corresponder à função hash do mecanismo de assinatura.

Valores válidos:

- mgf1-sha1

- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Desencapsular uma chave com o RSA-PKCS usando a CloudHSM CLI

Use o comando `key unwrap rsa-pkcs` na CloudHSM CLI para desencapsular uma chave de carga útil usando a chave privada RSA e o mecanismo de desencapsulamento RSA-PKCS.

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que elas não foram geradas localmente, o atributo `local` é definido como `false`.

Para usar o `unwrap rsa-pkcs` comando `key`, você deve ter a chave privada RSA da chave de empacotamento pública RSA em seu AWS CloudHSM cluster e seu `unwrap` atributo deve estar definido como `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key unwrap rsa-pkcs
```

```
Usage: key unwrap rsa-pkcs [OPTIONS] --filter [<FILTER>...] --key-type-  
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--filter [<FILTER>...]
```

Key reference (e.g. `key-reference=0xabc`) or space separated list of key attributes in the form of `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` to select a key to unwrap with

```
--data-path <DATA_PATH>
```

Path to the binary file containing the wrapped key data

```
--data <DATA>
```

Base64 encoded wrapped key data

```
--attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
```

Space separated list of key attributes in the form of `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` for the unwrapped key

```
--share-crypto-users [<SHARE_CRYPTO_USERS;...]
```

Space separated list of Crypto User usernames to share the unwrapped key with

```
--manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
```

The quorum value for key management operations for the unwrapped key

```
--use-key-quorum-value <USE_KEY_QUORUM_VALUE;
```

The quorum value for key usage operations for the unwrapped key

```

--key-type-class <KEY_TYPE_CLASS>
    Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
--label <LABEL>
    Label for the unwrapped key
--session
    Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
--approval <APPROVAL>
    Filepath of signed quorum token file to approve operation
-h, --help
    Print help

```

Exemplos

Esses exemplos mostram como usar o comando `key unwrap rsa-oaep` usando uma chave AES com o valor do atributo `unwrap` definido como `true`.

Example Exemplo: desencapsular uma chave de carga útil dos dados da chave encapsulada codificada em Base64

```

aws-cloudhsm > key unwrap rsa-pkcs --key-type-class aes --label
aes-unwrapped --filter attr.label=rsa-private-key-example --data
am0Nc7+YE8FWs+5HvU7sIBcXVb24QA0165nbNAD+1bK+e18BpSfnaI3P+r8Dp+pLu1ofoUy/
vtzRjZoCiDofcz4EqCFnG14GdcJ1/3W/5WRvMatCa2d7cx02swaeZcjKsermPXYR01lG1fq6NskwMeeTkV8R7Rx9artFrs1
c3XdFJ2+0Bo94c6og/
yfPcp00obJlITCoXhtMRepSd040ggYq/6nUDuHCtJ86pPGnNahyr7+sAaSI3a5ECQLUjwaIARUCyoRh7EFK3qPXcg==
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08ef",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        }
      }
    }
  }
}

```



```

    },
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "aes",
    "label": "aes-unwrapped",
    "id": "0x",
    "check-value": "0x8d9099",
    "class": "secret-key",
    "encrypt": false,
    "decrypt": false,
    "token": true,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": false,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": true,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 16
  }
}
}
}
}

```

Example Exemplo: desencapsular uma chave de carga útil fornecida por meio de um caminho de dados

```

aws-cloudhsm > key unwrap rsa-pkcs --key-type-class aes --label aes-unwrapped --filter
attr.label=rsa-private-key-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x000000000001c08ef",

```

```
"key-info": {
  "key-owners": [
    {
      "username": "cu1",
      "key-coverage": "full"
    }
  ],
  "shared-users": [],
  "key-quorum-values": {
    "manage-key-quorum-value": 0,
    "use-key-quorum-value": 0
  },
  "cluster-coverage": "full"
},
"attributes": {
  "key-type": "aes",
  "label": "aes-unwrapped",
  "id": "0x",
  "check-value": "0x8d9099",
  "class": "secret-key",
  "encrypt": false,
  "decrypt": false,
  "token": true,
  "always-sensitive": false,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": false,
  "modifiable": true,
  "never-extractable": false,
  "private": true,
  "sensitive": true,
  "sign": true,
  "trusted": false,
  "unwrap": false,
  "verify": true,
  "wrap": false,
  "wrap-with-trusted": false,
  "key-length-bytes": 16
}
}
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços contendo atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave para ser usada no desencapsulamento.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário que contém os dados da chave encapsulada.

Obrigatório: sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados de chaves encapsuladas codificados em Base64.

Obrigatório: sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço contendo atributos de chaves na forma de `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para a chave encapsulada.

Obrigatório: não

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: `aes`, `des3`, `ec-private`, `generic-secret`, `rsa-private`].

Obrigatório: Sim

<LABEL>

Rótulo da chave desencapsulada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de desempacotamento for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

O comando key wrap na CLI do CloudHSM

O comando key wrap na CLI do CloudHSM exporta uma cópia criptografada de uma chave privada simétrica ou assimétrica do módulo de segurança de hardware (HSM) para um arquivo. Ao executar o comando key wrap, você especifica duas coisas: a chave a ser exportada e o arquivo de saída. A chave a ser exportada é uma chave no HSM que criptografará (encapsulará) a chave que você deseja exportar.

O comando key wrap não remove a chave do HSM nem impede que você use a chave em operações criptográficas. É possível exportar a mesma chave várias vezes. Para importar a chave criptografada novamente no HSM, use [O comando key unwrap na CLI do CloudHSM](#). Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários com quem essa chave é compartilhada podem usá-la somente em operações criptográficas.

O comando key wrap oferece consiste nos seguintes subcomandos:

- [aes-gcm](#)
- [aes-no-pad](#)

- [aes-pkcs5-pad](#)
- [aes-zero-pad](#)
- [cloudhsm-aes-gcm](#)
- [rsa-aes](#)
- [rsa-oaep](#)
- [rsa-pkcs](#)

Encapsular uma chave com o AES-GCM usando a CloudHSM CLI

Use o comando `key wrap aes-gcm` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave AES no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento AES-GCM. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap aes-gcm` comando, primeiro você deve ter uma chave AES em seu AWS CloudHSM cluster. Você pode gerar uma chave AES para encapsular com o comando [Gerar uma chave AES simétrica com a CloudHSM CLI](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap aes-gcm  
Usage: key wrap aes-gcm [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-  
filter [<WRAPPING_FILTER>...] --tag-length-bits <TAG_LENGTH_BITS>
```

Options:

--cluster-id *<CLUSTER_ID>*

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--payload-filter [*<PAYLOAD_FILTER>...*]

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a payload key

--wrapping-filter [*<WRAPPING_FILTER>...*]

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a wrapping key

--path *<PATH>*

Path to the binary file where the wrapped key data will be saved

--wrapping-approval *<WRAPPING_APPROVAL>*

File path of signed quorum token file to approve operation for wrapping key

--payload-approval *<PAYLOAD_APPROVAL>*

File path of signed quorum token file to approve operation for payload key

--aad *<AAD>*

Aes GCM Additional Authenticated Data (AAD) value, in hex

--tag-length-bits *<TAG_LENGTH_BITS>*

Aes GCM tag length in bits

-h, --help

Print help

Exemplo

Este exemplo mostra como usar o comando key wrap aes-gcm usando uma chave AES.

Example

```
aws-cloudhsm > key wrap aes-gcm --payload-filter attr.label=payload-key --wrapping-
filter attr.label=aes-example --tag-length-bits 64 --aad 0x10
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x000000000001c08f1",
    "wrapping_key_reference": "0x000000000001c08ea",
    "iv": "0xf90613bb8e337ec0339aad21",
    "wrapped_key_data": "xvslgrtg8kHzirvekny97tLSIeokpPwV8"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de encapsulamento.

Obrigatório: Sim

<AAD>

Valor de dados autenticados adicionais (AAD) do AES GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Comprimento do rótulo AES GCM em bits.

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Envolva uma chave AES-NO-PAD usando a CLI do CloudHSM

Use o comando `key wrap aes-no-pad` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave AES no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento AES-NO-PAD. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap aes-no-pad` comando, primeiro você deve ter uma chave AES em seu AWS CloudHSM cluster. Você pode gerar uma chave AES para encapsular usando o comando [Gerar uma chave AES simétrica com a CloudHSM CLI](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap aes-no-pad
```


Usage: key wrap aes-no-pad [OPTIONS] --payload-filter [**<PAYLOAD_FILTER>**...] --wrapping-filter [**<WRAPPING_FILTER>**...]

Options:

--cluster-id **<CLUSTER_ID>**

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--payload-filter [**<PAYLOAD_FILTER>**...]

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a payload key

--wrapping-filter [**<WRAPPING_FILTER>**...]

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a wrapping key

--path **<PATH>**

Path to the binary file where the wrapped key data will be saved

--wrapping-approval **<WRAPPING_APPROVALR>**

File path of signed quorum token file to approve operation for wrapping key

--payload-approval **<PAYLOAD_APPROVALR>**

File path of signed quorum token file to approve operation for payload key

-h, --help

Print help

Exemplo

Este exemplo mostra como usar o comando key wrap aes-no-pad usando uma chave AES com o valor do atributo wrap definido como true.

Example

```
aws-cloudhsm > key wrap aes-no-pad --payload-filter attr.label=payload-key --wrapping-
filter attr.label=aes-example
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x000000000001c08f1",
    "wrapping_key_reference": "0x000000000001c08ea",
    "wrapped_key_data": "eXK3PMA0nKM9y3YX6brbhtMoC060E0H9"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave de encapsulamento.

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando `key wrap` na CLI do CloudHSM](#)

- [O comando key unwrap na CLI do CloudHSM](#)

Encapsular uma chave com AES- PKCS5 -PAD usando o CloudHSM CLI

Use o comando `key wrap aes-pkcs5-pad` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave AES no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento AES-PKCS5-PAD. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap aes-pkcs5-pad` comando, primeiro você deve ter uma chave AES em seu AWS CloudHSM cluster. Você pode gerar uma chave AES para encapsular usando o comando [Gerar uma chave AES simétrica com a CloudHSM CLI](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap aes-pkcs5-pad
Usage: key wrap aes-pkcs5-pad [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --
wrapping-filter [<WRAPPING_FILTER>...]

Options:
  --cluster-id <CLUSTER_ID>
      Unique Id to choose which of the clusters in the config file to run the
      operation against. If not provided, will fall back to the value provided when
      interactive mode was started, or error
  --payload-filter [<PAYLOAD_FILTER>...]
      Key reference (e.g. key-reference=0xabc) or space separated list of key
      attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
      payload key
```

```

--wrapping-filter [<WRAPPING_FILTER>...]
    Key reference (e.g. key-reference=0xabc) or space separated list of key
    attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
    wrapping key
--path <PATH>
    Path to the binary file where the wrapped key data will be saved
--wrapping-approval <WRAPPING_APPROVALR>
    File path of signed quorum token file to approve operation for wrapping key
--payload-approval <PAYLOAD_APPROVALR>
    File path of signed quorum token file to approve operation for payload key
-h, --help
    Print help

```

Exemplo

Este exemplo mostra como usar o comando `key wrap aes-pkcs5-pad` usando uma chave AES com o valor do atributo `wrap` definido como `true`.

Example

```

aws-cloudhsm > key wrap aes-pkcs5-pad --payload-filter attr.label=payload-key --
wrapping-filter attr.label=aes-example
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x000000000001c08f1",
    "wrapping_key_reference": "0x000000000001c08ea",
    "wrapped_key_data": "MbuYNresf0KyGNxKWen88nSfX+uUE/0qmGofSisicY="
  }
}

```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de encapsulamento.

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando `key wrap` na CLI do CloudHSM](#)
- [O comando `key unwrap` na CLI do CloudHSM](#)

Envolva uma chave AES-ZERO-PAD usando a CLI do CloudHSM

Use o comando `key wrap aes-zero-pad` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave AES no módulo de segurança de hardware (HSM) e o mecanismo de encapsulamento AES-ZERO-PAD. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o key wrap aes-zero-pad comando, primeiro você deve ter uma chave AES em seu AWS CloudHSM cluster. Você pode gerar uma chave AES para encapsular usando o comando [Gerar uma chave AES simétrica com a CloudHSM CLI](#) com o atributo wrap definido como true.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap aes-zero-pad
```

```
Usage: key wrap aes-zero-pad [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --  
wrapping-filter [<WRAPPING_FILTER>...]
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--payload-filter [<PAYLOAD_FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a payload key

```
--wrapping-filter [<WRAPPING_FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a wrapping key

```
--path <PATH>
```

Path to the binary file where the wrapped key data will be saved

```
--wrapping-approval <WRAPPING_APPROVALR>
```

File path of signed quorum token file to approve operation for wrapping key

```
--payload-approval <PAYLOAD_APPROVALR>
```

File path of signed quorum token file to approve operation for payload key

```
-h, --help
```

Print help

Exemplo

Este exemplo mostra como usar o comando `key wrap aes-zero-pad` usando uma chave AES com o valor do atributo `wrap` definido como `true`.

Example

```
aws-cloudhsm > key wrap aes-zero-pad --payload-filter attr.label=payload-key --
wrapping-filter attr.label=aes-example
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x000000000001c08f1",
    "wrapping_key_reference": "0x000000000001c08ea",
    "wrapped_key_data": "L1wV1L/YeBNVAw6Mpk3owFJZXBzDL0Nt"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de encapsulamento.

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Envolva uma chave CLOUDHSM-AES-GCM usando a CLI do CloudHSM

Use o comando `key wrap cloudhsm-aes-gcm` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave AES no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento CLOUDHSM-AES-GCM. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap cloudhsm-aes-gcm` comando, primeiro você deve ter uma chave AES em seu AWS CloudHSM cluster. Você pode gerar uma chave AES para encapsular com o comando [Gerar uma chave AES simétrica com a CloudHSM CLI](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap cloudhsm-aes-gcm
Usage: key wrap cloudhsm-aes-gcm [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --
wrapping-filter [<WRAPPING_FILTER>...] --tag-length-bits <TAG_LENGTH_BITS>

Options:
  --cluster-id <CLUSTER_ID>
      Unique Id to choose which of the clusters in the config file to run the
      operation against. If not provided, will fall back to the value provided when
      interactive mode was started, or error
  --payload-filter [<PAYLOAD_FILTER>...]
      Key reference (e.g. key-reference=0xabc) or space separated list of key
      attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
      payload key
  --wrapping-filter [<WRAPPING_FILTER>...]
      Key reference (e.g. key-reference=0xabc) or space separated list of key
      attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
      wrapping key
  --path <PATH>
      Path to the binary file where the wrapped key data will be saved
  --wrapping-approval <WRAPPING_APPROVALR>
      File path of signed quorum token file to approve operation for wrapping key
  --payload-approval <PAYLOAD_APPROVALR>
      File path of signed quorum token file to approve operation for payload key
  --aad <AAD>
      Aes GCM Additional Authenticated Data (AAD) value, in hex
  --tag-length-bits <TAG_LENGTH_BITS>
      Aes GCM tag length in bits
  -h, --help
      Print help
```

Exemplo

Este exemplo mostra como usar o comando `key wrap cloudhsm-aes-gcm` usando uma chave AES.

Example

```
aws-cloudhsm > key wrap cloudhsm-aes-gcm --payload-filter attr.label=payload-key --
wrapping-filter attr.label=aes-example --tag-length-bits 64 --aad 0x10
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x000000000001c08f1",
    "wrapping_key_reference": "0x000000000001c08ea",
    "wrapped_key_data": "6Rn8nkjEriDYlnP3P8nPkyQ8hp10EJ899zsrF+aTB0i/fI1Z"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE selecionar uma chave de encapsulamento.

Obrigatório: Sim

<AAD>

Valor de dados autenticados adicionais (AAD) do AES GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Comprimento do rótulo AES GCM em bits.

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Criar uma chave com o RSA-AES usando a CloudHSM CLI

Use o comando `key wrap rsa-aes` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave pública RSA no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento RSA-AES. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap rsa-aes` comando, primeiro você deve ter uma chave RSA em seu AWS CloudHSM cluster. Você pode gerar um par de chaves RSA usando o comando [A generate-asymmetric-pair categoria na CLI do CloudHSM](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap rsa-aes
```

```
Usage: key wrap rsa-aes [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-  
filter [<WRAPPING_FILTER>...] --hash-function <HASH_FUNCTION> --mgf <MGF>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--payload-filter [<PAYLOAD_FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a payload key

```
--wrapping-filter [<WRAPPING_FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a wrapping key

```
--path <PATH>
```

Path to the binary file where the wrapped key data will be saved

```
--wrapping-approval <WRAPPING_APPROVALR>
```

File path of signed quorum token file to approve operation for wrapping key

```
--payload-approval <PAYLOAD_APPROVALR>
```

File path of signed quorum token file to approve operation for payload key

```
--hash-function <HASH_FUNCTION>
```

Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]

```
--mgf <MGF>
```

Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224, mgf1-sha256, mgf1-sha384, mgf1-sha512]

```
-h, --help
```

Print help

Exemplo

Este exemplo mostra como usar o comando `key wrap rsa-aes` usando uma chave pública RSA com o valor do atributo `wrap` definido como `true`.

Example

```
aws-cloudhsm > key wrap rsa-aes --payload-filter attr.label=payload-key --wrapping-
filter attr.label=rsa-public-key-example --hash-function sha256 --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "payload-key-reference": "0x000000000001c08f1",
    "wrapping-key-reference": "0x000000000007008da",
    "wrapped-key-data": "HrSE1DEyLjIeyGdPa9R+ebiqB5TIJGyamPker31ZebPwRA
+NcerbAJ08DJ11XPygZcI21vIFSZJuWMEiWpe1R9D/5WSYgxLVKex30xCFqebtEzxbKuv4D0mU4meSofqREYvtb3EoIKwjy
+RL5WGXXKe4nAboAkC5G07veI5yHL1SaKlssSJtTL/CFpbSLsAFuYbv/NUCWwMY5mwyVTCSlw+HlgKK
+5TH1MzBaSi8fpfyepLT8sHy2Q/VR16ifb49p6m0KQFbRVvz/OWUd6l4d97BdgtaEz6ueg=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de encapsulamento.

Obrigatório: Sim

<MGF>

Especifica a função de geração da máscara.

Note

A função hash da função de geração de máscara deve corresponder à função hash do mecanismo de assinatura.

Valores válidos

- `mgf1-sha1`
- `mgf1-sha224`
- `mgf1-sha256`
- `mgf1-sha384`
- `mgf1-sha512`

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Encapsular uma chave com o RSA-OAEP usando a CloudHSM CLI

Use o comando `key wrap rsa-oaep` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave pública RSA no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento RSA-OAEP. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap rsa-oaep` comando, primeiro você deve ter uma chave RSA em seu AWS CloudHSM cluster. Você pode gerar um par de chaves RSA usando o comando [A generate-asymmetric-pair categoria na CLI do CloudHSM](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap rsa-oaep
```

```
Usage: key wrap rsa-oaep [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-  
filter [<WRAPPING_FILTER>...] --hash-function <HASH_FUNCTION> --mgf <MGF>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--payload-filter [<PAYLOAD_FILTER>...]
    Key reference (e.g. key-reference=0xabc) or space separated list of key
    attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
    payload key
--wrapping-filter [<WRAPPING_FILTER>...]
    Key reference (e.g. key-reference=0xabc) or space separated list of key
    attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
    wrapping key
--path <PATH>
    Path to the binary file where the wrapped key data will be saved
--wrapping-approval <WRAPPING_APPROVALR>
    File path of signed quorum token file to approve operation for wrapping key
--payload-approval <PAYLOAD_APPROVALR>
    File path of signed quorum token file to approve operation for payload key
--hash-function <HASH_FUNCTION>
    Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
--mgf <MGF>
    Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224,
    mgf1-sha256, mgf1-sha384, mgf1-sha512]
-h, --help
    Print help
```

Exemplo

Este exemplo mostra como usar o comando `key wrap rsa-oaep` usando uma chave pública RSA com o valor do atributo `wrap` definido como `true`.

Example

```
aws-cloudhsm > key wrap rsa-oaep --payload-filter attr.label=payload-key --wrapping-
filter attr.label=rsa-public-key-example --hash-function sha256 --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "payload-key-reference": "0x000000000001c08f1",
    "wrapping-key-reference": "0x000000000007008da",
    "wrapped-key-data": "0jJe4msobPLz9TuSAdULEu17T5rMDWtS1LyBSkLbaZnYzzpdrhsbGLbwZJCtB/
jGkDNdB4qyTA0QwEpggGf6v+Yx6JcesNeKKNU8XZa1/YBoHC8noTGUSDI2qr+u2tDc84NPv6d
+F2K00NXsSxMhmxxzNG/gzTVIJh0uy/B1yHjGP4m0XoDZf5+7f5M1CjxBmz4Vva/
wrWHGCSG0y0aWblEv0iHAIIt3UBdyKmU+/
My4xjJfJv7WGGu3DFUUIZ06TihRtKQhUYU1M9u6NPf9riJJfHsk6QCUSZ9yWThDT9as6i7e3htnyDhIhGWaoK8JU855cN/
YNKAUqkNpC4FPL3iw=="
  }
}
```



```
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, `key-reference=0xabc`) ou lista separada por espaços de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para selecionar uma chave de encapsulamento.

Obrigatório: Sim

<MGF>

Especifica a função de geração da máscara.

Note

A função hash da função de geração de máscara deve corresponder à função hash do mecanismo de assinatura.

Valores válidos

- `mgf1-sha1`

- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Criar uma chave com o RSA-PKCS usando a CloudHSM CLI

Use o comando `key wrap rsa-pkcs` na CloudHSM CLI para encapsular uma chave de carga útil usando uma chave pública RSA no módulo de segurança de hardware (HSM) e no mecanismo de encapsulamento RSA-PKCS. O atributo `extractable` da chave de carga útil deve ser definido como `true`.

Somente o proprietário de uma chave, ou seja, o usuário de criptografia que a criou, pode encapsulá-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas.

Para usar o `key wrap rsa-pkcs` comando, primeiro você deve ter uma chave RSA em seu AWS CloudHSM cluster. Você pode gerar um par de chaves RSA usando o comando [A generate-asymmetric-pair categoria na CLI do CloudHSM](#) e o atributo `wrap` definido como `true`.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptomoedas (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado(a) como um CU.

Sintaxe

```
aws-cloudhsm > help key wrap rsa-pkcs
```

```
Usage: key wrap rsa-pkcs [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-  
filter [<WRAPPING_FILTER>...]
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--payload-filter [<PAYLOAD_FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a payload key

```
--wrapping-filter [<WRAPPING_FILTER>...]
```

Key reference (e.g. key-reference=0xabc) or space separated list of key attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a wrapping key

```
--path <PATH>
```

Path to the binary file where the wrapped key data will be saved

```
--wrapping-approval <WRAPPING_APPROVAL>
```

File path of signed quorum token file to approve operation for wrapping key

```
--payload-approval <PAYLOAD_APPROVAL>
```

File path of signed quorum token file to approve operation for payload key

```
-h, --help
```

Print help

Exemplo

Este exemplo mostra como usar o comando `key wrap rsa-pkcs` usando uma chave pública RSA.

Example

```
aws-cloudhsm > key wrap rsa-pkcs --payload-filter attr.label=payload-key --wrapping-  
filter attr.label=rsa-public-key-example
```

```
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x000000000001c08f1",
    "wrapping_key_reference": "0x000000000007008da",
    "wrapped_key_data": "am0Nc7+YE8FWs+5HvU7sIBcXVb24QA0l65nbNAD+1bK+e18BpSfnaI3P+r8Dp
+PLu1ofoUy/
vtzRjZoCiDofcz4EqCFnGl4GdcJ1/3W/5WRvMatCa2d7cx02swaeZcjKsermPXYR01lGlFq6NskwMeeTkV8R7Rx9artFrs1
c3XdFJ2+0Bo94c6og/
yfPcp00obJlITCoXhtMRepSd040ggYq/6nUDuHCtJ86pPGnNahyr7+sAaSI3a5ECQLUjwaIARUCyoRh7EFK3qPXcg=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PAYLOAD_FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE selecionar uma chave de carga útil.

Obrigatório: Sim

<PATH>

Caminho para o arquivo binário em que os dados da chave encapsulada serão salvos.

Obrigatório: não

<WRAPPING_FILTER>

Referência de chave (por exemplo, key-reference=0xabc) ou lista separada por espaços de atributos de chave na forma de attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE selecionar uma chave de encapsulamento.

Obrigatório: Sim

<WRAPPING_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de encapsulamento. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de encapsulamento for maior do que 1.

<PAYLOAD_APPROVALR>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação da chave de carga útil. Exigido somente se o valor do quórum do serviço de gerenciamento de chaves da chave de carga útil for maior do que 1.

Tópicos relacionados

- [O comando key wrap na CLI do CloudHSM](#)
- [O comando key unwrap na CLI do CloudHSM](#)

Fazer login em um HSM usando a CLI do CloudHSM

Você pode usar o login comando na CLI do CloudHSM para entrar e sair de cada segurança de hardware (HSM) em um cluster. AWS CloudHSM Esse comando tem o subcomando a seguir:

- [mfa-token-sign](#)

Note

Se você exceder cinco tentativas incorretas de login, sua conta será bloqueada. Para desbloquear a conta, um administrador deve redefinir sua senha usando o comando [user change-password](#) em cloudhsm_cli.

Para solucionar problemas de login e logout

Se você tiver mais de um HSM no cluster, pode ser que mais tentativas de login sejam permitidas antes de a sua conta ser bloqueada. Isso ocorre porque o cliente CloudHSM equilibra a carga em vários. HSMs Sendo assim, pode ser que as tentativas de login não comecem sempre no mesmo HSM. Se você estiver testando essa funcionalidade, recomendamos fazer isso em um cluster com apenas um HSM ativo.

Se você criou seu cluster antes de fevereiro de 2018, sua conta será bloqueada após 20 tentativas incorretas de login.

Tipo de usuário

Os usuários a seguir podem executar esses comandos.

- Administrador desativado
- Administrador
- Crypto user (CU) (usuário de criptografia)

Sintaxe

```
aws-cloudhsm > help login
```

Login to your cluster

USAGE:

```
cloudhsm-cli login [OPTIONS] --username <USERNAME> --role <ROLE> [COMMAND]
```

Commands:

```
mfa-token-sign  Login with token-sign mfa
help           Print this message or the help of the given subcommand(s)
```

OPTIONS:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--username <USERNAME>
```

Username to access the Cluster

```
--role <ROLE>
```

Role the user has in the Cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

```
--password <PASSWORD>
```

Optional: Plaintext user's password. If you do not include this argument you will be prompted for it

```
-h, --help
    Print help (see a summary with '-h')
```

Exemplo

Example

Esse comando conecta você a todos HSMs em um cluster com as credenciais de um usuário administrador chamado `admin1`.

```
aws-cloudhsm > login --username admin1 --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_). O nome de usuário não diferencia maiúsculas de minúsculas neste comando. O nome de usuário é sempre exibido em minúsculas.

Obrigatório: Sim

<ROLE>

Especifica a função atribuída a esse usuário. Os valores válidos são `admin`, `crypto-user`.

Para obter a função do usuário, use o `user list` comando. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Noções básicas dos usuários de HSM](#).

Obrigatório: Sim

<PASSWORD>

Especifica a senha do usuário que está fazendo login no HSMs.

Obrigatório: os clientes devem fornecer a senha por meio do argumento da linha de `--password` comando ou omiti-la para ser solicitada interativamente.

 Note

Usar o `--password` argumento na linha de comando pode expor as credenciais em seu histórico de shell. Considere solicitar a senha de forma interativa ou limpar o histórico do shell após o uso.

Tópicos relacionados

- [Conceitos básicos sobre a CloudHSM CLI](#)
- [Ativar o cluster](#)

Fazer login com MFA em um HSM usando a CloudHSM CLI

Use o `login mfa-token-sign` comando na CLI do AWS CloudHSM para fazer login em um módulo de segurança de hardware (HSM) usando autenticação multifator (MFA). Para usar esse comando, você precisa primeiro configurar a [MFA para a CloudHSM CLI](#).

Tipo de usuário

Os usuários a seguir podem executar esses comandos.

- Administrador
- Crypto user (CU) (usuário de criptografia)

Sintaxe

```
aws-cloudhsm > help login mfa-token-sign
```

```
Login with token-sign mfa
```

```
USAGE:
```

```
login --username <username> --role <role> mfa-token-sign --token <token>
```


OPTIONS:

`--cluster-id <CLUSTER_ID>` Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--token <TOKEN>` Filepath where the unsigned token file will be written

`-h, --help` Print help

Exemplo

Example

```
aws-cloudhsm > login --username test_user --role admin mfa-token-sign --token /home/valid.token
Enter password:
Enter signed token file path (press enter if same as the unsigned token file):
{
  "error_code": 0,
  "data": {
    "username": "test_user",
    "role": "admin"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<TOKEN>

Caminho onde o arquivo de token não assinado será gravado.

Obrigatório: Sim

Tópicos relacionados

- [Conceitos básicos sobre a CloudHSM CLI](#)
- [Ativar o cluster](#)
- [Usar a CloudHSM CLI para gerenciar a MFA](#)

Sair de um HSM usando a CLI do CloudHSM

Use o logout comando na CLI do CloudHSM para sair de cada módulo de segurança de hardware (HSM) em um cluster. AWS CloudHSM

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Crypto user (CU) (usuário de criptografia)

Sintaxe

```
aws-cloudhsm > help logout
Logout of your cluster

USAGE:
    logout

OPTIONS:
    --cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the
    config file to run the operation against. If not provided, will fall back to the value
    provided when interactive mode was started, or error
    -h, --help                    Print help information
    -V, --version                 Print version information
```

Exemplo

Example

Esse comando desconecta você de tudo HSMs em um cluster.

```
aws-cloudhsm > logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

Tópicos relacionados

- [Conceitos básicos sobre a CloudHSM CLI](#)

- [Ativar o cluster](#)

A categoria user na CLI do CloudHSM

Na CLI do CloudHSM, user é uma categoria principal para um grupo de comandos que, quando combinado com a categoria principal, cria um comando específico para usuários. Atualmente, a categoria de usuário consiste nos seguintes comandos:

- [user change-mfa](#)
- [alterar senha de usuário](#)
- [criar usuário](#)
- [excluir usuário](#)
- [lista de usuários](#)
- [replicação do usuário](#)

A categoria user change-mfa na CLI do CloudHSM

Na CLI do CloudHSM, user change-mfa é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando específico para mudar a autenticação multifator (MFA) para os usuários.

Atualmente, essa categoria consiste nos seguintes subcomandos:

- [sinal de token](#)

Alterar a configuração de MFA de um usuário com a CLI do CloudHSM

Use o comando user change-mfa token-sign na CLI do CloudHSM para atualizar a configuração de autenticação multifator (MFA) de uma conta de usuário. Qualquer conta de usuário pode executar esse comando. Contas com a função de administrador podem executar esse comando para outros usuários.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Usuário de criptografia

Sintaxe

Atualmente, há apenas uma única estratégia multifatorial disponível para os usuários: Token Sign.

```
aws-cloudhsm > help user change-mfa
```

Change a user's Mfa Strategy

Usage:

```
user change-mfa <COMMAND>
```

Commands:

```
token-sign  Register or Deregister a public key using token-sign mfa strategy
help        Print this message or the help of the given subcommand(s)
```

A estratégia Token Sign solicita um arquivo Token para gravar tokens não assinados.

```
aws-cloudhsm > help user change-mfa token-sign
```

Register or Deregister a public key using token-sign mfa strategy

```
Usage: user change-mfa token-sign [OPTIONS] --username <USERNAME> --role <ROLE> <--
token <TOKEN>|--deregister>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--username <USERNAME>
```

Username of the user that will be modified

```
--role <ROLE>
```

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

```
--change-password <CHANGE_PASSWORD>
```

Optional: Plaintext user's password. If you do not include this argument you will be prompted for it

```

--token <TOKEN>
    Filepath where the unsigned token file will be written. Required for enabling
    MFA for a user

--approval <APPROVAL>
    Filepath of signed quorum token file to approve operation

--deregister
    Deregister the MFA public key, if present

--change-quorum
    Change the Quorum public key along with the MFA key

-h, --help
    Print help (see a summary with '-h')
```

Exemplo

Esse comando grava um token não assinado por HSM em seu cluster no arquivo especificado por token. Quando for solicitado, assine os tokens no arquivo.

Example: grave um token não assinado por HSM em seu cluster

```

aws-cloudhsm > user change-mfa token-sign --username cu1 --change-password password --
role crypto-user --token /path/myfile
Enter signed token file path (press enter if same as the unsigned token file):
Enter public key PEM file path:/path/mypemfile
{
  "error_code": 0,
  "data": {
    "username": "test_user",
    "role": "admin"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<ROLE>

Especifica a função atribuída à conta do usuário. Esse parâmetro é obrigatório. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Noções básicas dos usuários de HSM](#).

Valores válidos

- Administrador: os administradores podem gerenciar usuários, mas não podem gerenciar chaves.
- Usuário de criptografia: usuários de criptografia podem criar chaves de gerenciamento e usar chaves em operações criptográficas.

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_).

Não é possível alterar o nome de um usuário após sua criação. Em comandos da CLI do CloudHSM, a função e a senha diferenciam maiúsculas de minúsculas, mas o nome do usuário não diferencia.

Obrigatório: sim

<CHANGE_PASSWORD>

Especifica a nova senha em texto simples do usuário cujo MFA está sendo registrado/cancelado.

Obrigatório: sim

<TOKEN>

Caminho onde o arquivo de token não assinado será gravado.

Obrigatório: sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de usuário do quórum for maior que 1.

<DEREGISTER>

Cancela o registro da chave pública da MFA, se presente.

<CHANGE-QUORUM>

Altera a chave pública do quórum junto com a chave de MFA.

Tópicos relacionados

- [Noções básicas sobre 2FA para usuários de HSM](#)

Alterar a senha de um usuário com a CLI do CloudHSM

Use o `user change-password` comando na CLI do CloudHSM para alterar a senha de um usuário existente em seu cluster. AWS CloudHSM Para habilitar o MFA para um usuário, use o comando `user change-mfa`.

Qualquer usuário pode alterar sua própria senha. Além disso, os usuários com a função de administrador podem alterar a senha de outro usuário no cluster. Você não precisa inserir a senha atual para fazer a alteração.

Note

No entanto, não é possível alterar a senha de um usuário que está conectado no momento ao cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Crypto user (CU) (usuário de criptografia)

Sintaxe

Note

Para habilitar a autenticação multifator (MFA) para um usuário, use o comando `user change-mfa`.

aws-cloudhsm > help user change-password

Change a user's password

Usage:

```
cloudhsm-cli user change-password [OPTIONS] --username <USERNAME> --role <ROLE>
[--password <PASSWORD>]
```

Options:

--cluster-id <CLUSTER_ID>

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--username <USERNAME>

Username of the user that will be modified

--role <ROLE>

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

--password <PASSWORD>

Optional: Plaintext user's password. If you do not include this argument you will be prompted for it

--approval <APPROVAL>

Filepath of signed quorum token file to approve operation

--deregister-mfa <DEREGISTER-MFA>

Deregister the user's mfa public key, if present

--deregister-quorum <DEREGISTER-QUORUM>

Deregister the user's quorum public key, if present

-h, --help

Print help (see a summary with '-h')

Exemplo

Os exemplos a seguir mostram como usar `user change-password` para redefinir a senha do usuário atual ou de qualquer outro usuário em seu cluster.

Example: Alterar sua senha

Qualquer usuário no cluster pode usar `user change-password` para alterar sua própria senha.

A saída a seguir mostra que Bob está atualmente conectado como um usuário de criptografia (CU).

```
aws-cloudhsm > user change-password --username bob --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "bob",
    "role": "crypto-user"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de usuário do quórum for maior que 1.

<DEREGISTER-MFA>

Cancela o registro da chave pública da MFA, se presente.

<DEREGISTER-QUORUM>

Cancele o registro da chave pública do quórum, se estiver presente.

<PASSWORD>

Especifica a nova senha em texto simples do usuário. Os seguintes caracteres não são aceitos ':'

Obrigatório: sim

<ROLE>

Especifica a função atribuída à conta do usuário. Esse parâmetro é obrigatório. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Noções básicas dos usuários de HSM](#).

Valores válidos

- Administrador: os administradores podem gerenciar usuários, mas não podem gerenciar chaves.
- Usuário de criptografia: usuários de criptografia podem criar chaves de gerenciamento e usar chaves em operações criptográficas.

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_).

Não é possível alterar o nome de um usuário após sua criação. Em comandos da CLI do CloudHSM, a função e a senha diferenciam maiúsculas de minúsculas, mas o nome do usuário não diferencia.

Obrigatório: sim

Tópicos relacionados

- [lista de usuários](#)
- [criar usuário](#)
- [excluir usuário](#)

A categoria de quórum de alterações de usuário na CLI do CloudHSM

Na CLI do CloudHSM, user change-quorum é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando específico para mudar o quórum para os usuários.

user change-quorum é usado para registrar a autenticação de quórum de usuários usando uma estratégia de quórum especificada. A partir do SDK 5.8.0, há apenas uma única estratégia de quórum disponível para os usuários, conforme mostrado abaixo.

Atualmente, essa categoria consiste nos seguintes categorias e subcomandos:

- [sinal de token](#)
 - [registro](#)

A categoria user change-quorum token-sign na CLI do CloudHSM

Na CLI do CloudHSM, user change-quorum token-sign é uma categoria principal para um grupo de comandos que, quando combinado com a categoria principal, cria um comando específico para operações de quórum de assinatura de token.

Atualmente, essa categoria consiste nos seguintes comandos:

- [registro](#)

Registrar a estratégia de quórum de assinatura de token de um usuário usando a CLI do CloudHSM

Use o comando user change-quorum token-sign register na CLI do CloudHSM para registrar a estratégia de quorum de assinatura de token para um usuário administrador.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador

Sintaxe

```
aws-cloudhsm > help user change-quorum token-sign register
```

Register a user for quorum authentication with a public key

Usage: user change-quorum token-sign register --public-key *<PUBLIC_KEY>* --signed-token *<SIGNED_TOKEN>*

Options:

--cluster-id <i><CLUSTER_ID></i>	Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error
--public-key <i><PUBLIC_KEY></i>	Filepath to public key PEM file
--signed-token <i><SIGNED_TOKEN></i>	Filepath with token signed by user private key

```
-h, --help Print help (see a summary with '-h')
```

Exemplo

Example

Para executar este comando, você precisará estar logado como o usuário que deseja register quorum token-sign.

```
aws-cloudhsm > login --username admin1 --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
}
```

O comando `user change-quorum token-sign register` registrará sua chave pública no HSM. Como resultado, ele o qualificará como aprovador de quórum para operações exigidas por quórum que exijam que o usuário obtenha assinaturas de quórum para atingir o limite de valor de quórum necessário.

```
aws-cloudhsm > user change-quorum token-sign register \
  --public-key /home/mypemfile \
  --signed-token /home/mysignedtoken
{
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
}
```

Agora você pode executar o comando `user list` e confirmar se o quorum token-sign foi registrado para esse usuário.

```
aws-cloudhsm > user list
{
  "error_code": 0,
```

```
"data": {
  "users": [
    {
      "username": "admin",
      "role": "admin",
      "locked": "false",
      "mfa": [],
      "quorum": [],
      "cluster-coverage": "full"
    },
    {
      "username": "admin1",
      "role": "admin",
      "locked": "false",
      "mfa": [],
      "quorum": [
        {
          "strategy": "token-sign",
          "status": "enabled"
        }
      ],
      "cluster-coverage": "full"
    }
  ]
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<PUBLIC-KEY>

Caminho de arquivo para o arquivo PEM de chave pública.

Obrigatório: sim

<SIGNED-TOKEN>

Caminho de arquivo com token assinado pela chave privada do usuário.

Obrigatório: sim

Tópicos relacionados

- [Usar a CLI do CloudHSM para gerenciar a autenticação de quórum](#)
- [Usar a autenticação de quórum para administradores: configuração inicial](#)
- [Altere o valor mínimo do quórum para administradores](#)
- [Nomes e tipos de serviços que oferecem suporte à autenticação de quórum](#)

Crie um AWS CloudHSM usuário com o CloudHSM CLI

O user create comando na CLI do CloudHSM cria um usuário no seu cluster. AWS CloudHSM Somente contas de usuário com a função de administrador podem executar esse comando.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Administrador

Requisitos

Para executar esse comando, você deve estar registrado em log como usuário administrador

Sintaxe

```
aws-cloudhsm > help user create
```

Create a new user

```
Usage: cloudhsm-cli user create [OPTIONS] --username <USERNAME> --role <ROLE> [--password <PASSWORD>]
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--username <USERNAME>
```

Username to access the HSM cluster

`--role <ROLE>`

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

`--password <PASSWORD>`

Optional: Plaintext user's password. If you do not include this argument you will be prompted for it

`--approval <APPROVAL>`

Filepath of signed quorum token file to approve operation

`-h, --help`

Print help (see a summary with '-h')

Exemplo

Esses exemplos mostram como usar `user create` para criar novos usuários em seus HSMs.

Example: criar um usuário de criptografia

Este exemplo cria uma conta em seu AWS CloudHSM cluster com a função de usuário criptográfico.

```
aws-cloudhsm > user create --username alice --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "alice",
    "role": "crypto-user"
  }
}
```

Argumentos

`<CLUSTER_ID>`

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_). O nome de usuário não diferencia maiúsculas de minúsculas neste comando. O nome de usuário é sempre exibido em minúsculas.

Obrigatório: Sim

<ROLE>

Especifica a função atribuída a esse usuário. Esse parâmetro é obrigatório. Os valores válidos são admin, crypto-user.

Para obter a função do usuário, use o comando user list. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Noções básicas dos usuários de HSM](#).

<PASSWORD>

Especifica a senha do usuário que está fazendo login no HSMs. Os seguintes caracteres não são aceitos ':'

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de usuário do quórum for maior que 1.

Tópicos relacionados

- [lista de usuários](#)
- [excluir usuário](#)
- [alterar senha de usuário](#)

Excluir um AWS CloudHSM usuário com a CLI do CloudHSM

O user delete comando na CLI do CloudHSM exclui um usuário do seu cluster. AWS CloudHSM Somente contas de usuário com a função de administrador podem executar esse comando. Você não pode excluir um usuário que esteja atualmente conectado a um HSM.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Administrador

Requisitos

- Você não pode excluir contas de usuários que possuem chaves.
- Sua conta de usuário deve ter a função de administrador para executar esse comando.

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
aws-cloudhsm > help user delete
```

Delete a user

```
Usage: user delete [OPTIONS] --username <USERNAME> --role <ROLE>
```

Options:

```
--cluster-id <CLUSTER_ID>
```

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

```
--username <USERNAME>
```

Username to access the HSM cluster

```
--role <ROLE>
```

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

```
--approval <APPROVAL>
```

Filepath of signed quorum token file to approve operation

Exemplo

```
aws-cloudhsm > user delete --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "username": "alice",
    "role": "crypto-user"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<USERNAME>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_). O nome de usuário não diferencia maiúsculas de minúsculas neste comando. O nome de usuário é sempre exibido em minúsculas.

Obrigatório: Sim

<ROLE>

Especifica a função atribuída a esse usuário. Esse parâmetro é obrigatório. Os valores válidos são admin, crypto-user.

Para obter a função do usuário, use o comando user list. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Noções básicas dos usuários de HSM](#).

Obrigatório: Sim

<APPROVAL>

Especifica o caminho do arquivo para um arquivo de token de quórum designado para aprovar a operação. Exigido somente se o valor do quórum do serviço de usuário do quórum for maior que 1.

Obrigatório: Sim

Tópicos relacionados

- [lista de usuários](#)
- [criar usuário](#)
- [alterar senha de usuário](#)

Listar todos os AWS CloudHSM usuários com o CloudHSM CLI

O `user list` comando na CLI do CloudHSM lista as contas de usuário presentes no seu cluster. AWS CloudHSM Você não precisa estar conectado na CLI do CloudHSM para executar esse comando.

Note

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração que o AWS CloudHSM cliente e as ferramentas de linha de comando usam. Caso contrário, as alterações feitas podem não ser efetivas HSMs em todas as partes do cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

```
aws-cloudhsm > help user list  
List the users in your cluster
```

USAGE:

```
user list
```

Options:

```
--cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the  
config file to run the operation against. If not provided, will fall back to the value  
provided when interactive mode was started, or error  
-h, --help Print help
```

Exemplo

Este comando lista os usuários presentes em seu cluster do CloudHSM.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "test_user",
        "role": "admin",
        "locked": "false",
        "mfa": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

A saída inclui os seguintes atributos do usuário:

- User Name: exibe o nome amigável definido pelo usuário para o usuário. O nome de usuário é sempre exibido em letras minúsculas.

- **Role:** determina as operações que o usuário pode executar no HSM.
- **Bloqueado:** indica se essa conta de usuário foi bloqueada.
- **MFA:** indica os mecanismos de autenticação multifatorial compatíveis com essa conta de usuário.
- **Cobertura do cluster:** indica a disponibilidade dessa conta de usuário em todo o cluster.

Tópicos relacionados

- [listUsers](#) em key_mgmt_util
- [criar usuário](#)
- [excluir usuário](#)
- [alterar senha de usuário](#)

Replicar um usuário com a CloudHSM CLI

Use o `user replicate` comando na CLI do CloudHSM para replicar um usuário de um cluster de origem para um cluster de destino AWS CloudHSM . AWS CloudHSM

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Administradores () COs

Requisitos

- Os clusters de origem e destino devem ser clones. Isso significa que um foi criado a partir de um backup do outro, ou ambos foram criados a partir de um backup comum. Consulte [Criar clusters de backups](#) para obter mais informações.
- Para executar esse comando, é necessário estar registrado como administrador nos clusters de origem e de destino.
 - No modo de comando único, o comando usará as variáveis ambientais `CLOUDHSM_PIN` e `CLOUDHSM_ROLE` para se autenticar no cluster de origem. Consulte [Modo de comando único](#) para obter mais informações. Para fornecer credenciais ao cluster de destino, é preciso definir duas variáveis ambientais adicionais: `DESTINATION_CLOUDHSM_PIN` e `DESTINATION_CLOUDHSM_ROLE`:

```
$ export DESTINATION_CLOUDHSM_ROLE=<role>
```

```
$ export DESTINATION_CLOUDHSM_PIN=<username:password>
```

- No modo interativo, os usuários precisarão fazer login explícito nos clusters de origem e de destino.

Sintaxe

```
aws-cloudhsm > help user replicate
```

Replicate a user from a source to a destination cluster

Usage: user replicate --username <USERNAME> --role <ROLE> --source-cluster-id <SOURCE_CLUSTER_ID> --destination-cluster-id <DESTINATION_CLUSTER_ID>

Options:

--username <USERNAME>

Username of the user to replicate

--role <ROLE>

Role the user has in the cluster

Possible values:

- crypto-user: A CryptoUser has the ability to manage and use keys
- admin: An Admin has the ability to manage user accounts

--source-cluster-id <SOURCE_CLUSTER_ID>

Source cluster ID

--destination-cluster-id <DESTINATION_CLUSTER_ID>

Destination cluster ID

-h, --help

Print help (see a summary with '-h')

Exemplos

Example Exemplo: replicar usuário

Esse comando replica um usuário de um cluster de origem para um cluster de destino clonado. O exemplo abaixo demonstra a saída quando conectado como um administrador em ambos os clusters.

```
admin-user@cluster-1234abcdefg > user replicate \  
  --username example-admin \  
  --role admin \  
  --source-cluster-id cluster-1234abcdefg \  
  --destination-cluster-id cluster-2345bcdefgh  
{  
  "error_code": 0,  
  "data": {  
    "user": {  
      "username": "example-admin",  
      "role": "admin",  
      "locked": "false",  
      "mfa": [],  
      "quorum": [],  
      "cluster-coverage": "full"  
    },  
    "message": "Successfully replicated user"  
  }  
}
```

Argumentos

<USERNAME>

Especifica o nome de usuário do usuário a ser replicado no cluster de origem.

Obrigatório: Sim

<ROLE>

Especifica a função atribuída a esse usuário. Esse parâmetro é obrigatório. Os valores válidos são admin, crypto-user.

Para obter a função do usuário, use o comando user list. Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Noções básicas dos usuários de HSM](#).

Obrigatório: Sim

<SOURCE_CLUSTER_ID>

O ID do cluster de origem.

Obrigatório: Sim

<DESTINATION_CLUSTER_ID>

O ID do cluster de destino.

Obrigatório: Sim

Tópicos relacionados

- [Conectar-se a vários clusters com a CLI do CloudHSM](#)

A categoria de quórum na CloudHSM CLI

Na CloudHSM CLI, quorum é uma categoria principal para um grupo de comandos que, quando combinado com quorum, cria um comando específico para autenticação de quórum ou operações M de N. Atualmente, essa categoria consiste na subcategoria token-sign que consiste em seus próprios comandos. Clique no link a seguir para obter detalhes.

- [sinal de token](#)

Serviços administrativos: a autenticação de quórum é usada para serviços com privilégios administrativos, como criar usuários, excluir usuários, alterar senhas de usuários, definir valores de quórum e desativar recursos de quórum e MFA.

Serviços de criptografia para usuários: a autenticação de quórum é usada para serviços privilegiados de usuários criptográficos associados a uma chave específica, como assinar com uma chave, uma chave, sharing/unsharing uma chave e definir o atributo de wrapping/unwrapping uma chave. O valor do quórum de uma chave associada é configurado quando a chave é gerada, importada ou desencapsulada. O valor do quórum deve ser igual ou menor do que o número de usuários aos quais a chave está associada, o que inclui usuários com os quais a chave é compartilhada e o proprietário da chave.

Cada tipo de serviço é subdividido em um nome de serviço qualificado, que contém um conjunto específico de operações de serviço suportadas por quórum que podem ser executadas.

Nome do serviço	Tipo de serviço	Operações de serviço
usuário	Administrador	- criar usuário - excluir usuário - alterar senha de usuário - user change-mfa
quorum	Administrador	sinal simbólico de quórum set-quorum-value
cluster ¹	Administrador	- cluster mtls register-trust-anchor - cluster mtls deregister-trust-anchor - cluster mtls set-enforcement
gerenciamento de chaves	Usuário de criptografia	- encapsulamento de chaves - desencapsulamento de chaves - compartilhamento de chaves - descompartilhamento de chaves - key set-attribute
uso de chave	Usuário de criptografia	sinal de chave

[1] O serviço de cluster está disponível exclusivamente em instâncias hsm2m.medium

Tópicos relacionados

- [Configurar a autenticação de quórum para AWS CloudHSM administradores usando o CloudHSM CLI](#)
- [Gerenciar a autenticação de quórum \(controle de acesso M ou N\) usando a CLI do CloudHSM](#)

A categoria quorum token-sign na CLI do CloudHSM

Na CLI do CloudHSM, quorum token-sign é uma categoria para um grupo de comandos que, quando combinado com quorum token-sign, cria um comando específico para autenticação de quórum ou operações M de N.

Atualmente, essa categoria consiste nos seguintes comandos:

- [excluir](#)
- [generate](#)
- [listar](#)
- [list-quorum-values](#)
- [set-quorum-value](#)

Excluir tokens de quórum usando a CloudHSM CLI

Use o comando do quorum token-sign delete na CloudHSM CLI para excluir um ou mais tokens de um serviço autorizado por quórum.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador

Sintaxe

```
aws-cloudhsm > help quorum token-sign delete
```

```
Delete one or more Quorum Tokens
```

```
Usage: quorum token-sign delete --scope <SCOPE>
```

```
Options:
```

```
--cluster-id <CLUSTER_ID>
```

```
    Unique Id to choose which of the clusters in the config file to run the  
    operation against. If not provided, will fall back to the value provided when  
    interactive mode was started, or error
```

```
--scope <SCOPE>
```

Scope of which token(s) will be deleted

Possible values:

- user: Deletes all token(s) of currently logged in user
- all: Deletes all token(s) on the HSM

-h, --help

Print help (see a summary with '-h')

Exemplo

O exemplo seguinte mostra como o comando do quorum token-sign delete na CloudHSM CLI pode ser usado para excluir um ou mais tokens de um serviço autorizado por quórum.

Example: exclui um ou mais tokens de um serviço autorizado por quórum

```
aws-cloudhsm > quorum token-sign delete --scope all
{
  "error_code": 0,
  "data": "Deletion of quorum token(s) successful"
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<SCOPE>

O escopo no qual os tokens serão excluídos no AWS CloudHSM cluster.

Valores válidos

- Usuário: usado para excluir somente tokens pertencentes ao usuário conectado.
- Todos: usado para excluir todos os tokens no AWS CloudHSM cluster.

Tópicos relacionados

- [lista de usuários](#)
- [criar usuário](#)

- [excluir usuário](#)

Gerar um token de quórum usando a CloudHSM CLI

Use o comando `quorum token-sign generate` na CloudHSM CLI para gerar um token para um serviço autorizado de quórum.

Há um limite para obter um token ativo por usuário por serviço em um cluster HSM para usuários e quórum de serviços. Esse limite não se aplica ao tokens relacionados aos serviços chave.

Note

Somente Administradores e Usuários de criptografia podem gerar tokens de serviço específicos. Para obter mais informações sobre tipos e nomes de serviços, consulte [Nomes e tipos de serviços que oferecem suporte à autenticação de quórum](#).

Serviços administrativos: a autenticação de quórum é usada para serviços com privilégios administrativos, como criar usuários, excluir usuários, alterar senhas de usuários, definir valores de quórum e desativar recursos de quórum e MFA.

Serviços de criptografia para usuários: a autenticação de quórum é usada para serviços privilegiados de usuários criptográficos associados a uma chave específica, como assinar com uma chave, uma chave, sharing/unsharing uma chave e definir o atributo de wrapping/unwrapping uma chave. O valor do quórum de uma chave associada é configurado quando a chave é gerada, importada ou desencapsulada. O valor do quórum deve ser igual ou menor do que o número de usuários aos quais a chave está associada, o que inclui usuários com os quais a chave é compartilhada e o proprietário da chave.

Cada tipo de serviço é subdividido em um nome de serviço qualificado, que contém um conjunto específico de operações de serviço suportadas por quórum que podem ser executadas.

Nome do serviço	Tipo de serviço	Operações de serviço
usuário	Administrador	• criar usuário • excluir usuário • alterar senha de usuário • user change-mfa

Nome do serviço	Tipo de serviço	Operações de serviço
quorum	Administrador	• sinal simbólico de quórum set-quorum-value
cluster ¹	Administrador	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gerenciamento de chaves	Usuário de criptografia	• encapsulamento de chaves • desencapsulamento de chaves • compartilhamento de chaves • descompartilhamento de chaves • key set-attribute
uso de chave	Usuário de criptografia	• sinal de chave

[1] O serviço de cluster está disponível exclusivamente em instâncias hsm2m.medium

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Crypto user (CU) (usuário de criptografia)

Sintaxe

```
aws-cloudhsm > help quorum token-sign generate
Generate a token
```

```
Usage: quorum token-sign generate --service <SERVICE> --token <TOKEN>
```

Options:

`--cluster-id <CLUSTER_ID>`

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

`--service <SERVICE>`

Service the token will be used for

Possible values:

- user:

User management service is used for executing quorum authenticated user management operations

- quorum:

Quorum management service is used for setting quorum values for any quorum service

- cluster:

Cluster management service is used for executing quorum for cluster wide configuration managements like mtls enforcement, mtls registration and mtls deregistration

- registration:

Registration service is used for registering a public key for quorum authentication

- key-usage:

Key usage service is used for executing quorum authenticated key usage operations

- key-management:

Key management service is used for executing quorum authenticated key management operations

`--token <TOKEN>`

Filepath where the unsigned token file will be written

`-h, --help`

Print help

Exemplo

Esse comando grava um token não assinado por HSM em seu cluster no arquivo especificado por token.

Example: grave um token não assinado por HSM em seu cluster

```
aws-cloudhsm > quorum token-sign generate --service user --token /home/tfile
```

```
{
  "error_code": 0,
  "data": {
    "filepath": "/home/tfile"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<SERVICE>

Especifica o serviço autorizado de quórum para o qual gerar um token. Esse parâmetro é obrigatório.

Valores válidos

- user: serviço de gerenciamento de usuário usado para executar operações de gerenciamento de usuários autorizados por quórum.
- quorum: serviço de gerenciamento de quórum usado para definir valores de quórum autorizado para qualquer serviço autorizado por quórum.
- cluster: o serviço de gerenciamento de cluster usado para executar o quórum para gerenciamentos de configuração em todo o cluster, como imposição de mtls, registro de mtls e cancelamento de registro de mtls.
- registration: gera um token não assinado para uso no registro de uma chave pública para autorização de quórum.
- key-usage: gera um token não assinado que é usado para executar operações de uso de chaves autorizadas por quórum.
- key-management: gera um token não assinado que é usado para executar operações de gerenciamento de chaves autorizadas por quórum.

Obrigatório: sim

<TOKEN>

Caminho onde o arquivo de token não assinado será gravado.

Obrigatório: sim

Tópicos relacionados

- [Nomes e tipos de serviços que oferecem suporte à autenticação de quórum](#)

Listar tokens de quórum usando a CloudHSM CLI

Use o quorum token-sign list comando na CLI do CloudHSM para listar todos os tokens de quorum de assinatura de token presentes em seu cluster. AWS CloudHSM Isso inclui tokens gerados por outros usuários. Um token está vinculado a um usuário e, portanto, embora seja possível possa ver tokens de outros usuários, você poderá usar somente os tokens associados ao usuário atualmente conectado.

Para obter mais informações sobre tipos e nomes de serviços, consulte [Nomes e tipos de serviços que oferecem suporte à autenticação de quorum](#). Para saber mais sobre o conteúdo exibido nos tokens listados, consulte [the section called “Gerenciamento e uso de chaves com quórum \(M de N\)”](#) para os tokens associados aos serviços key-management e key-usage e consulte [the section called “Gerenciamento de usuários com quórum \(M de N\)”](#) para os tokens associados aos serviços user, quorum ou cluster, respectivamente.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Crypto user (CU) (usuário de criptografia)

Sintaxe

```
aws-cloudhsm > help quorum token-sign list  
List the token-sign tokens in your cluster
```

```
Usage: quorum token-sign list
```

```
Options:
```

```
    --cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the  
    config file to run the operation against. If not provided, will fall back to the value  
    provided when interactive mode was started, or error
```


`-h, --help``Print help`

Exemplo

Esse comando listará todos os tokens de assinatura de token presentes em seu cluster. AWS CloudHSM Isso inclui tokens gerados por outros usuários. Um token está vinculado a um usuário e, portanto, embora seja possível possa ver tokens de outros usuários, você poderá usar somente os tokens associados ao usuário atualmente conectado.

Example

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": [
      {
        "username": "admin",
        "service": "quorum",
        "approvals-required": 2,
        "number-of-approvals": 0,
        "token-timeout-seconds": 397,
        "cluster-coverage": "full"
      },
      {
        "username": "admin",
        "service": "user",
        "approvals-required": 2,
        "number-of-approvals": 0,
        "token-timeout-seconds": 588,
        "cluster-coverage": "full"
      },
      {
        "username": "crypto_user1",
        "service": "key-management",
        "key-reference": "0x000000000002c33f7",
        "minimum-token-count": 1
      },
      {
        "username": "crypto_user1",
        "service": "key-usage",
        "key-reference": "0x000000000002c33f7",
        "minimum-token-count": 1
      }
    ]
  }
}
```

```
}  
  ]  
}  
}
```

Tópicos relacionados

- [quorum token-sign generate](#)

Mostrar valores de quórum usando a CLI do CloudHSM

Use o `quorum token-sign list-quorum-values` comando na CLI do CloudHSM para listar os valores de quorum definidos no seu cluster. AWS CloudHSM

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

```
aws-cloudhsm > help quorum token-sign list-quorum-values
```

```
List current quorum values
```

```
Usage: quorum token-sign list-quorum-values
```

```
Options:
```

```
--cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the  
config file to run the operation against. If not provided, will fall back to the value  
provided when interactive mode was started, or error
```

```
-h, --help Print help
```

Exemplo

Esse comando lista os valores de quórum definidos em seu AWS CloudHSM cluster para cada serviço.

Example

hsm1.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 1
  }
}
```

hsm2m.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 1,
    "cluster": 1
  }
}
```

Tópicos relacionados

- [Nomes e tipos de serviços que oferecem suporte à autenticação de quórum](#)
- [Configurar mTLS \(recomendado\)](#)

Atualizar um valor de quórum usando a CLI do CloudHSM

Use o comando `quorum token-sign set-quorum-value` na CLI do CloudHSM para definir um novo valor de quórum para um serviço autorizado de quórum.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador

Sintaxe

```
aws-cloudhsm > help quorum token-sign set-quorum-value
```

Set a quorum value

Usage: quorum token-sign set-quorum-value [OPTIONS] --service *<SERVICE>* --value *<VALUE>*

Options:

--cluster-id *<CLUSTER_ID>*

Unique Id to choose which of the clusters in the config file to run the operation against. If not provided, will fall back to the value provided when interactive mode was started, or error

--service *<SERVICE>*

Service the token will be used for

Possible values:

- user:

User management service is used for executing quorum authenticated user management operations

- quorum:

Quorum management service is used for setting quorum values for any quorum service

- cluster:

Cluster management service is used for executing quorum for cluster wide configuration managements like mtls enforcement, mtls registration and mtls deregistration

--value *<VALUE>*

Value to set for service

--approval *<APPROVAL>*

Filepath of signed quorum token file to approve operation

-h, --help

Print help (see a summary with '-h')

Exemplo

Example

No exemplo a seguir, esse comando grava um token não assinado por HSM em seu cluster no arquivo especificado pelo token. Quando for solicitado, assine os tokens no arquivo.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value 2
{
```

```
"error_code": 0,  
"data": "Set Quorum Value successful"  
}
```

Em seguida, você pode executar o comando `list-quorum-values` para confirmar se o valor do quórum para o serviço de gerenciamento de quórum foi definido:

hsm1.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values  
{  
  "error_code": 0,  
  "data": {  
    "user": 1,  
    "quorum": 2  
  }  
}
```

hsm2m.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values  
{  
  "error_code": 0,  
  "data": {  
    "user": 1,  
    "quorum": 2,  
    "cluster": 1  
  }  
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster em que essa operação será executada.

Obrigatório: se vários clusters tiverem sido [configurados](#).

<APPROVAL>

O caminho do arquivo de token assinado a ser aprovado no HSM.

<SERVICE>

Especifica o serviço autorizado de quórum para o qual gerar um token. Esse parâmetro é obrigatório. Para obter mais informações sobre tipos e nomes de serviços, consulte [Nomes e tipos de serviços que oferecem suporte à autenticação de quorum](#).

Valores válidos

- usuário: o serviço de gerenciamento de usuários. Serviço usado para executar operações de gerenciamento de usuários autorizados por quórum.
- quorum: o serviço de gerenciamento de quórum. Serviço usado para definir valores de quórum autorizado para qualquer serviço autorizado por quórum.
- cluster: o serviço de gerenciamento de cluster usado para executar o quórum para gerenciamentos de configuração em todo o cluster, como imposição de mtls, registro de mtls e cancelamento de registro de mtls.
- registro: gera um token não assinado para uso no registro de uma chave pública para autorização de quórum.

Obrigatório: sim

<VALUE>

Especifica o valor do quorum a ser definido. O valor máximo do quórum é oito (8).

Obrigatório: sim

Tópicos relacionados

- [sinal simbólico de quórum list-quorum-values](#)
- [Nomes e tipos de serviços que oferecem suporte à autenticação de quórum](#)
- [Configurar mTLS \(recomendado\)](#)

AWS CloudHSM Utilitário de gerenciamento (CMU)

A ferramenta de linha de comando `cloudhsm_mgmt_util` ajuda os agentes de criptografia a gerenciar usuários nos módulos de segurança de hardware () em clusters. HSMs AWS CloudHSM O Utilitário de AWS CloudHSM Gerenciamento (CMU) inclui ferramentas que criam, excluem e listam usuários e alteram as senhas dos usuários.

O CMU e o Key Management Utility (KMU) fazem parte do [pacote do Client SDK 3](#). O Client SDK 3 e suas ferramentas de linha de comando relacionadas (Key Management Utility e CloudHSM Management Utility) estão disponíveis somente no tipo hsm1.medium do HSM.

cloudhsm_mgmt_util também inclui comandos que permitem que usuários criptográficos (CUs) compartilhem chaves e obtenham e definam atributos de chave. Esses comandos complementam os comandos de gerenciamento de chaves na ferramenta de gerenciamento de chave primária, [key_mgmt_util](#).

Para começar rapidamente, consulte [Clusters clonados em AWS CloudHSM](#). Para obter informações detalhadas sobre comandos da cloudhsm_mgmt_util e exemplos de uso dos comandos, consulte [Referência para comandos AWS CloudHSM do Management Utility](#).

Tópicos

- [Plataformas compatíveis com o AWS CloudHSM Management Utility](#)
- [Introdução ao AWS CloudHSM Management Utility \(CMU\)](#)
- [Instale e configure o AWS CloudHSM cliente para CMU \(Linux\)](#)
- [Instale e configure o AWS CloudHSM cliente para CMU \(Windows\)](#)
- [Referência para comandos AWS CloudHSM do Management Utility](#)

Plataformas compatíveis com o AWS CloudHSM Management Utility

Este tópico descreve as plataformas Linux e Windows suportadas pelo AWS CloudHSM Management Utility (CMU).

Suporte a Linux

- Amazon Linux
- Amazon Linux 2
- CentOS 6.10+
- CentOS 7.3+
- CentOS 8
- Red Hat Enterprise Linux (RHEL) 6.10+
- Red Hat Enterprise Linux (RHEL) 7.9+

- Red Hat Enterprise Linux (RHEL) 8
- Ubuntu 16.04 LTS
- Ubuntu 18.04 LTS

Suporte ao Windows

- Microsoft Windows Server 2012
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Introdução ao AWS CloudHSM Management Utility (CMU)

AWS CloudHSM O Management Utility (CMU) permite gerenciar usuários do módulo de segurança de hardware (HSM). Use este tópico para iniciar as tarefas básicas de gerenciamento de usuários do HSM, como criar usuários, listar usuários e conectar a CMU ao cluster.

1. Para usar o CMU, você deve primeiro usar a ferramenta configure para atualizar a configuração local do CMU com o `--cmu` parâmetro e um endereço IP de um dos do seu HSMs cluster. Faça isso sempre que usar o CMU para garantir que você está gerenciando usuários do HSM em cada HSM do cluster.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Digite o seguinte comando para iniciar o CLI no modo interativo.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```


Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

A saída deve ser semelhante à seguinte, dependendo de quantas HSMs você tem.

```
Connecting to the server(s), it may take time
depending on the server(s) load, please wait...

Connecting to server '10.0.2.9': hostname '10.0.2.9', port 2225...
Connected to server '10.0.2.9': hostname '10.0.2.9', port 2225.

Connecting to server '10.0.3.11': hostname '10.0.3.11', port 2225...
Connected to server '10.0.3.11': hostname '10.0.3.11', port 2225.

Connecting to server '10.0.1.12': hostname '10.0.1.12', port 2225...
Connected to server '10.0.1.12': hostname '10.0.1.12', port 2225.
```

O prompt muda para `aws-cloudhsm>` quando `cloudhsm_mgmt_util` está em execução.

3. Use o comando `loginHSM` para fazer login no cluster. Qualquer usuário de qualquer tipo pode usar esse comando para fazer login no cluster.

O comando no exemplo a seguir faz login do admin, que é o [responsável pela criptografia \(CO\)](#), por padrão. A senha desse usuário foi definida quando ativou o cluster. É possível usar o parâmetro `-hpswd` para ocultar sua senha.

```
aws-cloudhsm>loginHSM C0 admin -hpswd
```

O sistema solicita que você forneça sua senha. Você insere a senha, o sistema oculta a senha e a saída mostra que o comando foi bem-sucedido e que você se conectou a todos os HSMs do cluster.

```
Enter password:

loginHSM success on server 0(10.0.2.9)
loginHSM success on server 1(10.0.3.11)
```

```
loginHSM success on server 2(10.0.1.12)
```

4. Use `listUsers` para listar todos os usuários no cluster.

```
aws-cloudhsm>listUsers
```

O CMU lista todos os usuários no cluster.

Users on server 0(10.0.2.9):

Number of users found:2

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	C0	admin	N0
	0		N0
2	AU	app_user	N0
	0		N0

Users on server 1(10.0.3.11):

Number of users found:2

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	C0	admin	N0
	0		N0
2	AU	app_user	N0
	0		N0

Users on server 2(10.0.1.12):

Number of users found:2

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	C0	admin	N0
	0		N0
2	AU	app_user	N0
	0		N0

5. Use `createUser` para criar um usuário de CU chamado **example_user** com uma senha de **password1**.

Você usa usuários de CU em seus aplicativos para realizar operações criptográficas e de gerenciamento de chaves. É possível criar usuários de CU porque, na etapa 3, você fez login

como usuário CO. Somente usuários CO podem realizar tarefas de gerenciamento de usuários com CMU, como criar e excluir usuários e alterar as senhas de outros usuários.

```
aws-cloudhsm>createUser CU example_user password1
```

O CMU avisa sobre a operação de criação de usuário.

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?
```

6. Para criar o usuário de CU **example_user**, digite **y**.
7. Use `listUsers` para listar todos os usuários no cluster.

```
aws-cloudhsm>listUsers
```

O CMU lista todos os usuários no cluster, incluindo o novo usuário de CU que você acabou de criar.

```
Users on server 0(10.0.2.9):
Number of users found:3

  User Id      User Type      User Name      NO
MofnPubKey    LoginFailureCnt 2FA
    1          0          CO          admin
    2          0          AU          app_user
    3          0          CU          example_user
    0          0          NO
```

```
Users on server 1(10.0.3.11):
Number of users found:3
```

User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	C0	admin	N0
	0		N0
2	AU	app_user	N0
	0		N0
3	CU	example_user	N0
	0		N0
Users on server 2(10.0.1.12):			
Number of users found:3			
User Id	User Type	User Name	
MofnPubKey	LoginFailureCnt	2FA	
1	C0	admin	N0
	0		N0
2	AU	app_user	N0
	0		N0
3	CU	example_user	N0
	0		N0

8. Use o `logoutHSM` comando para sair do HSMs.

```
aws-cloudhsm>logoutHSM
```

```
logoutHSM success on server 0(10.0.2.9)
logoutHSM success on server 1(10.0.3.11)
logoutHSM success on server 2(10.0.1.12)
```

9. Use o comando `quit` para encerrar `cloudhsm_mgmt_util`.

```
aws-cloudhsm>quit
```

```
disconnecting from servers, please wait...
```

Instale e configure o AWS CloudHSM cliente para CMU (Linux)

Para interagir com o módulo de segurança de hardware (HSM) em seu AWS CloudHSM cluster usando o `cloudhsm_mgmt_util` (CMU), você precisa do software cliente para Linux. AWS CloudHSM Você deve instalá-lo na instância do cliente do Linux Amazon EC2 criada anteriormente. Você

também pode instalar um cliente se estiver usando o Windows. Para obter mais informações, consulte [Instale e configure o AWS CloudHSM cliente para CMU \(Windows\)](#).

Tarefas

- [Etapa 1. Instale as ferramentas AWS CloudHSM do cliente e da linha de comando](#)
- [Etapa 2. Editar a configuração do cliente](#)

Etapa 1. Instale as ferramentas AWS CloudHSM do cliente e da linha de comando

Conecte-se à sua instância cliente e execute os comandos a seguir para baixar e instalar as ferramentas do AWS CloudHSM cliente e da linha de comando.

Amazon Linux

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

RHEL 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

RHEL 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client_latest_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client_latest_u18.04_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_u18.04_amd64.deb
```

Etapa 2. Editar a configuração do cliente

Antes de usar o AWS CloudHSM cliente para se conectar ao seu cluster, você deve editar a configuração do cliente.

Para editar a configuração do cliente

1. Se estiver instalando o Client SDK 3 em cloudhsm_mgmt_util, conclua as etapas a seguir para garantir que todos os nós no cluster estejam sincronizados.
 - a. Executar `configure -a <IP of one of the HSMs>`.
 - b. Reinicie o serviço do cliente.
 - c. Executar `configure -m`.
2. Copie o certificado de emissão, [aquele usado para assinar o certificado do cluster](#), para o seguinte local na instância do cliente: `/opt/cloudhsm/etc/customerCA.crt`. É necessário ter permissões de usuário raiz da instância do cliente para copiar o certificado para este local.
3. Use o comando [configure](#) a seguir para atualizar os arquivos de configuração das ferramentas do AWS CloudHSM cliente e da linha de comando, especificando o endereço IP do HSM em seu cluster. Para obter o endereço IP do HSM, visualize seu cluster no [AWS CloudHSM console](#) ou execute o [describe-clusters](#) AWS CLI comando. Na saída do comando, o endereço IP do HSM é o valor que está no campo `EniIp`. Se você tiver mais de um HSM, escolha o endereço IP de qualquer um deles HSMs; não importa qual.

```
sudo /opt/cloudhsm/bin/configure -a <IP address>
```

```
Updating server config in /opt/cloudhsm/etc/cloudhsm_client.cfg
```

```
Updating server config in /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

4. Acesse [Ative o cluster em AWS CloudHSM](#).

Instale e configure o AWS CloudHSM cliente para CMU (Windows)

Para trabalhar com um módulo de segurança de hardware (HSM) em seu AWS CloudHSM cluster no Windows usando o cloudhsm_mgmt_util (CMU), você precisa do software cliente para Windows. AWS CloudHSM Você deve instalá-lo na instância do Windows Server criada anteriormente.

Note

- Se você estiver atualizando o cliente, os arquivos de configuração existentes de instalações anteriores não serão substituídos.
- O instalador do AWS CloudHSM cliente para Windows registra automaticamente a API de criptografia: próxima geração (CNG) e o provedor de armazenamento de chaves (KSP). Para desinstalar o cliente, execute o instalador novamente e siga as instruções de desinstalação.
- Você também pode instalar um cliente se estiver usando o Linux. Para obter mais informações, consulte [Instale e configure o AWS CloudHSM cliente para CMU \(Linux\)](#).

Para instalar (ou atualizar) o cliente e as ferramentas da linha de comando mais recentes do Windows

1. Conecte-se à sua instância do Windows Server.
2. Baixe o [AWSCloudHSMClient-latestinstalador.msi](#).
3. Se estiver instalando o Client SDK 3 em cloudhsm_mgmt_util, conclua as etapas a seguir para garantir que todos os nós no cluster estejam sincronizados.
 - a. Executar `configure.exe -a <IP of one of the HSMs>`.
 - b. Reinicie o serviço do cliente.
 - c. Executar `configure.exe -m`.
4. Vá até o local de download e execute o instalador (AWSCloudHSMClient-latest.msi) com privilégios administrativos.
5. Siga as instruções do instalador e escolha Fechar após a conclusão do instalador.
6. Copie o certificado de emissão autoassinado, [aquele usado para assinar o certificado do cluster](#) para a pasta `C:\ProgramData\Amazon\CloudHSM`.

7. Execute o seguinte comando para atualizar os arquivos de configuração. Certifique-se de parar e iniciar o cliente durante a reconfiguração, se você estiver atualizando-o:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM IP address>
```

8. Acesse [Ative o cluster em AWS CloudHSM](#).

Referência para comandos AWS CloudHSM do Management Utility

A ferramenta de linha de comando AWS CloudHSM `cloudhsm_mgmt_util` ajuda os agentes de criptografia a gerenciar usuários nos módulos de segurança de hardware () no cluster. HSMs AWS CloudHSM Também inclui comandos que permitem que usuários criptográficos (CUs) compartilhem chaves e obtenham e definam atributos de chave. Esses comandos complementam os comandos de gerenciamento de chave primária na ferramenta de linha de comando [key_mgmt_util](#).

Para começar rapidamente, consulte [Clusters clonados em AWS CloudHSM](#).

Antes de executar um comando `cloudhsm_mgmt_util`, você deve iniciar `cloudhsm_mgmt_util` fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Para listar todos os comandos da `cloudhsm_mgmt_util`, execute o seguinte comando:

```
aws-cloudhsm> help
```

Para obter a sintaxe de um comando da `cloudhsm_mgmt_util`, execute o seguinte comando:

```
aws-cloudhsm> help <command-name>
```

Note

Use a sintaxe de acordo com a documentação. Embora a ajuda integrada do software possa fornecer opções adicionais, essas não devem ser consideradas compatíveis e não devem ser utilizadas no código de produção.

Para executar um comando, digite o nome do comando ou uma parte suficiente do nome para diferenciá-lo dos nomes de outros comandos da `cloudhsm_mgmt_util`.

Por exemplo, para obter uma lista de usuários no HSMs, digite `listUsers` ou `listU`.

```
aws-cloudhsm> listUsers
```

Para encerrar sua sessão da `cloudhsm_mgmt_util`, execute o seguinte comando:

```
aws-cloudhsm> quit
```

Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Os seguintes tópicos descrevem comandos na `cloudhsm_mgmt_util`.

Note

Alguns comandos no `key_mgmt_util` e `cloudhsm_mgmt_util` têm os mesmos nomes. No entanto, os comandos normalmente têm sintaxe diferente, saída diferente e funcionalidade ligeiramente diferente.

Command	Description	Tipo de usuário
changePswd	Altera as senhas dos usuários no HSMs. Qualquer usuário pode alterar sua própria senha. COs pode alterar a senha de qualquer pessoa.	CO
createUser	Cria usuários de todos os tipos no HSMs.	CO
deleteUser	Exclui usuários de todos os tipos do HSMs.	CO
findAllKeys	Obtém as chaves que um usuário possui ou compartilha. Também obtém um hash de propriedade da chave e	CO, AU

Command	Description	Tipo de usuário
	compartilhamento de dados para todas as chaves em cada HSM.	
getAttribute	Obtém um valor de atributo para uma AWS CloudHSM chave e o grava em um arquivo ou stdout (saída padrão).	CU
getHSMInfo	Obtém informações sobre o hardware no qual um HSM está sendo executado.	Tudo. O login não é necessário.
getKeyInfo	Obtém proprietários, usuários compartilhados e o status de autenticação de quorum de uma chave.	Tudo. O login não é necessário.
info	Obtém informações sobre um HSM, incluindo o endereço IP, o nome do host, a porta e o usuário atual.	Tudo. O login não é necessário.
listUsers	Obtém os usuários em cada um dos HSMs, seu tipo e ID de usuário e outros atributos.	Tudo. O login não é necessário.
loginHSM e logoutHSM	Fazer login e logout de um HSM.	Tudo.
sair	Sai de cloudhsm_mgmt_util.	Tudo. O login não é necessário.
servidor	Entra e sai do modo de servidor em um HSM.	Tudo.

Command	Description	Tipo de usuário
<u>registerQuorumPubChave</u>	Associa um usuário do HSM a um par de chaves RSA-2048 assimétrico.	CO
<u>setAttribute</u>	Altera o valor dos atributos de rótulo, criptografia, descrição, encapsulamento e desencapsulamento de uma chave existente.	CU
<u>shareKey</u>	Compartilha uma chave existente com outros usuários.	CU
<u>syncKey</u>	Sincroniza uma chave em clusters clonados AWS CloudHSM .	CU, CO
<u>syncUser</u>	Sincroniza um usuário em clusters clonados AWS CloudHSM .	CO

Alterar a senha de um usuário usando o CMU

Use o `changePswd` comando no AWS CloudHSM `cloudhsm_mgmt_util` (CMU) para alterar a senha de um usuário existente nos módulos de segurança de hardware (HSM) no cluster. AWS CloudHSM

Qualquer usuário pode alterar sua própria senha. Além disso, os oficiais de criptografia (COs e PCOs) podem alterar a senha de outro CO ou usuário criptográfico (CU). Você não precisa inserir a senha atual para fazer a alteração.

Note

Você não pode alterar a senha de um usuário que está atualmente conectado ao AWS CloudHSM cliente ou ao `key_mgmt_util`.

Para solucionar problemas do ChangePSWD

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Responsáveis pela criptografia (CO)
- Usuários de criptografia (CU)

Sintaxe

Insira os argumentos na ordem especificada no diagrama de sintaxe. Use o parâmetro `-hpswd` para mascarar sua senha. Para habilitar a autenticação de dois fatores (2FA) para um usuário CO, use o parâmetro `-2fa` e inclua um caminho de arquivo. Para obter mais informações, consulte [the section called “Argumentos”](#).

```
changePswd <user-type> <user-name> <password> [-hpswd] [-2fa </path/to/authdata>]
```

Exemplos

Os exemplos a seguir mostram como usar `changePassword` para redefinir a senha do usuário atual ou de qualquer outro usuário em seu HSMs.

Example: Alterar sua senha

Qualquer usuário do HSMs pode usar `changePswd` para alterar sua própria senha. Antes de alterar a senha, use [info](#) para obter informações sobre cada uma delas HSMs no cluster, incluindo o nome de usuário e o tipo de usuário do usuário conectado.

A saída a seguir mostra que Bob está atualmente conectado como um usuário de criptografia (CU).

```
aws-cloudhsm> info server 0
```

```

Id      Name      Hostname      Port  State      Partition
LoginState
0      10.1.9.193    10.1.9.193    2225  Connected  hsm-jqici4covtv
Logged in as 'bob(CU)'

```

```
aws-cloudhsm> info server 1
```

```

Id      Name      Hostname      Port  State      Partition
LoginState
1      10.1.10.7    10.1.10.7    2225  Connected  hsm-ogi3sywxbqx
Logged in as 'bob(CU)'

```

Para alterar a senha, Bob executa `changePswd` seguido pelo tipo e o nome de usuário e uma nova senha.

```
aws-cloudhsm> changePswd CU bob newPassword
```

```

*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

```

```

Do you want to continue(y/n)?y
Changing password for bob(CU) on 2 nodes

```

Example: Alterar a senha de outro usuário

Você deve ser CO ou PCO para alterar a senha de outro CO ou UC no. HSMs Antes de alterar a senha de outro usuário, use o comando [info](#) para confirmar se o tipo de usuário é CO ou PCO.

A saída a seguir confirma que Alice, que é uma CO, está conectada no momento.

```
aws-cloudhsm>info server 0
```

```

Id      Name      Hostname      Port  State      Partition
LoginState

```

```
0      10.1.9.193      10.1.9.193      2225  Connected      hsm-jqici4covtv
Logged in as 'alice(C0)'
```



```
aws-cloudhsm>info server 1
```

Id	Name	Hostname	Port	State	Partition
0	10.1.10.7	10.1.10.7	2225	Connected	hsm-ogi3sywxbqx

```
Logged in as 'alice(C0)'
```

Alice deseja redefinir a senha de outro usuário, John. Antes de alterar a senha, ela usa o comando [listUsers](#) para verificar o tipo de usuário de John.

A saída a seguir relaciona John como um usuário CO.

```
aws-cloudhsm> listUsers
Users on server 0(10.1.9.193):
Number of users found:5
```

User Id	User Type	User Name	MofnPubKey	
1	PC0	admin	YES	0
2	AU	jane	NO	0
3	CU	bob	NO	0
4	CU	alice	NO	0
5	CO	john	NO	0

```
Users on server 1(10.1.10.7):
Number of users found:5
```

User Id	User Type	User Name	MofnPubKey	
1	PC0	admin	YES	0
2	AU	jane	NO	0

3	CU	bob	NO	0
NO				
4	CO	alice	NO	0
NO				
5	CO	john	NO	0
NO				

Para alterar a senha, Alice executa `changePswd` seguido pelo tipo, o nome e uma nova senha para John.

```
aws-cloudhsm>changePswd CO john newPassword
```

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?y
Changing password for john(CO) on 2 nodes
```

Argumentos

Insira os argumentos na ordem especificada no diagrama de sintaxe. Use o parâmetro `-hpswd` para mascarar sua senha. Para habilitar 2FA para um usuário CO, use o parâmetro `-2fa` e inclua um caminho de arquivo. Para obter mais informações sobre como trabalhar com 2FA, consulte [Gerenciar a 2FA para usuários](#)

```
changePswd <user-type> <user-name> <password> [-hpswd] [-2fa </path/to/authdata>]
```

<user-type>

Especifica o tipo atual do usuário cuja senha você está mudando. Você não pode usar `changePswd` para alterar o tipo do usuário.

Os valores válidos são CO, CU, PCO e PRECO.

Para obter o tipo de usuário, use [listUsers](#). Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Tipos de usuário do HSM para o AWS CloudHSM Management Utility](#).

Obrigatório: Sim

<user-name>

Especifica o nome amigável do usuário. Esse parâmetro não diferencia maiúsculas de minúsculas. Você não pode usar changePswd para alterar o nome do usuário.

Obrigatório: Sim

<password | -hpswd >

Especifica uma nova senha para o usuário. Insira uma string de 7 a 32 caracteres. Esse valor diferencia maiúsculas de minúsculas. A senha aparece em texto sem formatação quando você a digita. Para ocultar sua senha, use o parâmetro -hpswd no lugar da senha e siga as instruções.

Obrigatório: Sim

[-2fa </ >path/to/authdata]

Especifica a ativação de 2FA para esse usuário de CO. Para obter os dados necessários para configurar 2FA, inclua um caminho para um local no sistema de arquivos com um nome de arquivo após o parâmetro -2fa. Para obter mais informações sobre como trabalhar com 2FA, consulte [Gerenciar a 2FA para usuários](#).

Obrigatório: não

Tópicos relacionados

- [info](#)
- [listUsers](#)
- [createUser](#)
- [deleteUser](#)

Crie um AWS CloudHSM usuário com CMU

Use o createUser comando em cloudhsm_mgmt_util (CMU) para criar um usuário nos módulos de segurança de hardware (HSM) no cluster. AWS CloudHSM Somente oficiais de criptografia (COs e PRECOs) podem executar esse comando. Quando o comando é bem-sucedido, ele cria o usuário em tudo HSMs no cluster.

Para solucionar problemas de createUser

Se a configuração do HSM for imprecisa, talvez o usuário não tenha sido criado em todos. HSMs Para adicionar o usuário a qualquer HSMs um que esteja ausente, use o comando [SyncUser](#) ou CreateUser somente HSMs nos que estão faltando esse usuário. Para evitar erros de configuração, execute a ferramenta [configure](#) com a opção -m.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Responsáveis pela criptografia (CO, PRECO)

Sintaxe

Insira os argumentos na ordem especificada no diagrama de sintaxe. Use o parâmetro -hpswd para mascarar sua senha. Para criar um usuário CO com autenticação de dois fatores (2FA), use o parâmetro -2fa e inclua um caminho de arquivo. Para obter mais informações, consulte [the section called “Argumentos”](#).

```
createUser <user-type> <user-name> <password> | -hpswd> [-2fa </path/to/authdata>]
```

Exemplos

Esses exemplos mostram como usar createUser para criar novos usuários em seu HSMs.

Example: Criar um responsável pela criptografia

Este exemplo cria um oficial de criptografia (CO) HSMs em um cluster. O primeiro comando usa [loginHSM](#) para fazer login no HSM como um oficial de criptografia.

```
aws-cloudhsm> loginHSM CO admin 735782961
```

```
loginHSM success on server 0(10.0.0.1)
loginHSM success on server 1(10.0.0.2)
loginHSM success on server 1(10.0.0.3)
```

O segundo comando usa o comando `createUser` para criar `alice`, um novo responsável pela criptografia no HSM.

A mensagem de advertência explica que o comando cria usuários HSMs em todos os do cluster. Mas, se o comando falhar em algum HSMs, o usuário não existirá neles HSMs. Para continuar, digite `y`.

A saída mostra que o novo usuário foi criado em todos os três HSMs no cluster.

```
aws-cloudhsm> createUser C0 alice 391019314

*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?Invalid option, please type 'y' or 'n'

Do you want to continue(y/n)?y
Creating User alice(C0) on 3 nodes
```

Quando o comando é concluído, `alice` tem as mesmas permissões no HSM que o usuário `admin C0`, incluindo a alteração da senha de qualquer usuário no. HSMs

O comando final usa o comando [ListUsers](#) para verificar se `alice` existe em todos os três HSMs no cluster. A saída também mostra que `alice` é atribuído ao ID de usuário 3.. Você usa o ID do usuário para se identificar `alice` em outros comandos, como [findAllKeys](#).

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:3
```

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		

```

1          PRECO          admin          YES
0          NO
2          AU            app_user        NO
0          NO
3          CO            alice           NO
0          NO

```

Users on server 1(10.0.0.2):

Number of users found:3

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PRECO	admin	YES
0	NO		
2	AU	app_user	NO
0	NO		
3	CO	alice	NO
0	NO		

Users on server 1(10.0.0.3):

Number of users found:3

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PRECO	admin	YES
0	NO		
2	AU	app_user	NO
0	NO		
3	CO	alice	NO
0	NO		

Example: Criar um usuário de criptografia

Este exemplo cria um usuário de criptografia (CU), bob, no HSM. Os usuários de criptografia podem criar e gerenciar chaves, mas não conseguem gerenciar usuários.

Depois de digitar y para responder à mensagem de advertência, a saída mostra que bob foi criada HSMs em todos os três no cluster. O novo CU pode fazer login no HSM para criar e gerenciar chaves.

O comando usava um valor de senha de defaultPassword. Mais tarde, bob ou qualquer CO pode usar o comando [changePswd](#) para alterar sua senha.

```
aws-cloudhsm> createUser CU bob defaultPassword
```

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?Invalid option, please type 'y' or 'n'

Do you want to continue(y/n)?y
Creating User bob(CU) on 3 nodes
```

Argumentos

Insira os argumentos na ordem especificada no diagrama de sintaxe. Use o parâmetro `-hpswd` para mascarar sua senha. Para criar um usuário CO com 2FA ativado, use o parâmetro `-2fa` e inclua um caminho de arquivo. Para obter mais informações sobre 2FA, consulte [Gerenciar a 2FA para usuários](#).

```
createUser <user-type> <user-name> <password> | -hpswd> [-2fa </path/to/authdata>]
```

<user-type>

Especifica o tipo de usuário. Esse parâmetro é obrigatório.

Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Tipos de usuário do HSM para o AWS CloudHSM Management Utility](#).

Valores válidos:

- CO: Oficiais de criptografia podem gerenciar usuários, mas não conseguem gerenciar chaves.
- CU: Usuários de criptografia podem criar chaves de gerenciamento e usar chaves em operações criptográficas.

O PRECO é convertido em um CO quando você atribui uma senha durante a [ativação do HSM](#).

Obrigatório: Sim

<user-name>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (`_`).

Não é possível alterar o nome de um usuário após sua criação. Em comandos da `cloudhsm_mgmt_util`, o tipo de usuário e a senha diferenciam maiúsculas de minúsculas, mas o nome do usuário não diferencia.

Obrigatório: Sim

<password | -hpswd >

Especifica uma senha para o usuário. Insira uma string de 7 a 32 caracteres. Esse valor diferencia maiúsculas de minúsculas. A senha aparece em texto sem formatação quando você a digita. Para ocultar sua senha, use o parâmetro `-hpswd` no lugar da senha e siga as instruções.

Para alterar uma senha de usuário, use [changePswd](#). Qualquer usuário do HSM pode alterar sua própria senha, mas os usuários de CO podem alterar a senha de qualquer usuário (de qualquer tipo) no HSMs.

Obrigatório: Sim

[-2fa </ >path/to/authdata]

Especifica a criação de um usuário CO com 2FA habilitado. Para obter os dados necessários para configurar a autenticação 2FA, inclua um caminho para um local no sistema de arquivos com um nome de arquivo após o parâmetro `-2fa`. Para obter mais informações sobre como configurar e trabalhar com 2FA, consulte [Gerenciar a 2FA para usuários](#).

Obrigatório: não

Tópicos relacionados

- [listUsers](#)
- [deleteUser](#)
- [syncUser](#)
- [changePswd](#)

Excluir um AWS CloudHSM usuário usando CMU

Use o `deleteUser` comando no AWS CloudHSM `cloudhsm_mgmt_util` (CMU) para excluir um usuário dos módulos de segurança de hardware (HSM) no cluster. AWS CloudHSM Somente responsáveis pela criptografia (COs) podem executar esse comando. Você não pode excluir um usuário que

esteja atualmente conectado a um HSM. Para obter mais informações sobre a exclusão de usuários, consulte [Como excluir usuários do HSM](#).

Tip

Você não pode excluir usuários de criptografia (CU) que possuem chaves.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- CO

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
deleteUser <user-type> <user-name>
```

Exemplo

Este exemplo exclui um oficial de criptografia (CO) do HSMs em um cluster. O primeiro comando usa [listUsers](#) para listar todos os usuários no. HSMs

A saída mostra que o usuário 3alice,, é um CO no HSMs.

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:3
```

User Id	User Type	User Name	MofnPubKey
1	PCO	admin	YES
0	N0		
2	AU	app_user	N0
0	N0		
3	CO	alice	N0
0	N0		

```
Users on server 1(10.0.0.2):
Number of users found:3
```

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PC0	admin	YES
0	N0		
2	AU	app_user	N0
0	N0		
3	C0	alice	N0
0	N0		

Users on server 1(10.0.0.3):

Number of users found:3

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PC0	admin	YES
0	N0		
2	AU	app_user	N0
0	N0		
3	C0	alice	N0
0	N0		

O segundo comando usa o `deleteUser` comando para excluir `alice` do HSMs.

A saída mostra que o comando foi bem-sucedido em todos os três HSMs no cluster.

```
aws-cloudhsm> deleteUser C0 alice
Deleting user alice(C0) on 3 nodes
deleteUser success on server 0(10.0.0.1)
deleteUser success on server 0(10.0.0.2)
deleteUser success on server 0(10.0.0.3)
```

O comando final usa o `listUsers` comando para verificar `alice` se ele foi excluído de todos os três HSMs no cluster.

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:2
```

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PC0	admin	YES
0	N0		

2	AU	app_user	NO
0	NO		
Users on server 1(10.0.0.2):			
Number of users found:2			
User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PCO	admin	YES
0	NO		
2	AU	app_user	NO
0	NO		
Users on server 1(10.0.0.3):			
Number of users found:2			
User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PCO	admin	YES
0	NO		
2	AU	app_user	NO
0	NO		

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
deleteUser <user-type> <user-name>
```

<user-type>

Especifica o tipo de usuário. Esse parâmetro é obrigatório.

Tip

Você não pode excluir usuários de criptografia (CU) que possuem chaves.

Os valores válidos são CO, CU.

Para obter o tipo de usuário, use [listUsers](#). Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Tipos de usuário do HSM para o AWS CloudHSM Management Utility](#).

Obrigatório: Sim

<user-name>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_).

Não é possível alterar o nome de um usuário após sua criação. Em comandos da `cloudhsm_mgmt_util`, o tipo de usuário e a senha diferenciam maiúsculas de minúsculas, mas o nome do usuário não diferencia.

Obrigatório: Sim

Tópicos relacionados

- [listUsers](#)
- [createUser](#)
- [syncUser](#)
- [changePswd](#)

Listar as chaves que um usuário AWS CloudHSM criptográfico possui usando CMU

Use o `findAllKeys` comando no AWS CloudHSM `cloudhsm_mgmt_util` (CMU) para obter as chaves que um usuário criptográfico (UC) especificado possui ou compartilha. AWS CloudHSM O comando também retorna um hash dos dados do usuário em cada um dos HSMs. Você pode usar o hash para determinar rapidamente se os dados de usuários, propriedade da chave e compartilhamento de chaves são iguais HSMs em todos no cluster. Na saída, as chaves de propriedade do usuário são anotadas por (o) e chaves compartilhadas são anotadas por (s).

`findAllKeys` retorna chaves públicas somente quando a UC especificada possui a chave, mesmo que todos CUs no HSM possam usar qualquer chave pública. Esse comportamento é diferente de [findKey](#) `key_mgmt_util`, que retorna as chaves públicas para todos os usuários CU.

Somente agentes de criptografia (COs e PCOs) e usuários de dispositivos (AUs) podem executar esse comando. Os usuários do Crypto (CUs) podem executar os seguintes comandos:

- [listUsers](#) para encontrar todos os usuários
- [findKey](#) em `key_mgmt_util` para encontrar as chaves que eles podem usar

- [getKeyInfo](#) em `key_mgmt_util` para encontrar o proprietário e os usuários compartilhados de uma chave específica que eles possuem ou compartilham

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Oficiais de criptografia (CO, PCO)
- Usuários de dispositivos (AU)

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
findAllKeys <user id> <key hash (0/1)> [<output file>]
```

Exemplos

Esses exemplos mostram como usar `findAllKeys` para encontrar todas as chaves de um usuário e obter um hash das principais informações do usuário em cada uma das HSMs.

Example: Encontrar as chaves para um CU

Este exemplo é usado `findAllKeys` para encontrar as chaves HSMs que o usuário 4 possui e compartilha. O comando usa um valor de 0 para o segundo argumento para suprimir o valor do hash. Como ele omite o nome do arquivo opcional, o comando grava em stdout (saída padrão).

A saída mostra que o usuário 4 pode usar 6 chaves: 8, 9, 17, 262162, 19 e 31. A saída usa um (s) para indicar chaves que são explicitamente compartilhadas pelo usuário. As chaves possuídas pelo usuário são indicadas por um (o) e incluem chaves simétricas e privadas que o usuário não compartilha e chaves públicas que estão disponíveis para todos os usuários de criptografia.

```
aws-cloudhsm> findAllKeys 4 0
Keys on server 0(10.0.0.1):
Number of keys found 6
number of keys matched from start index 0::6
8(s),9(s),17,262162(s),19(o),31(o)
findAllKeys success on server 0(10.0.0.1)

Keys on server 1(10.0.0.2):
Number of keys found 6
number of keys matched from start index 0::6
8(s),9(s),17,262162(s),19(o),31(o)
findAllKeys success on server 1(10.0.0.2)

Keys on server 1(10.0.0.3):
Number of keys found 6
number of keys matched from start index 0::6
8(s),9(s),17,262162(s),19(o),31(o)
findAllKeys success on server 1(10.0.0.3)
```

Example: verificar se os dados do usuário estão sincronizados

Este exemplo é usado `findAllKeys` para verificar se todos os itens HSMs do cluster contêm os mesmos usuários, propriedade da chave e valores de compartilhamento da chave. Para fazer isso, ele obtém um hash dos dados do usuário chave em cada HSM e compara os valores de hash.

Para obter o hash da chave, o comando usa um valor de 1 no segundo argumento. O nome do arquivo opcional é omitido e, portanto, o comando grava o hash da chave em stdout.

O exemplo especifica o usuário6, mas o valor do hash será o mesmo para qualquer usuário que possua ou compartilhe qualquer uma das chaves no. HSMs Se o usuário especificado não possui ou compartilha chaves, como um CO, o comando não retornará um valor de hash.

A saída mostra que o hash da chave é idêntico aos dois HSMs no cluster. Se um dos HSM tivesse usuários diferentes, diferentes proprietários de chaves ou diferentes usuários compartilhados, os valores de hash de chave não seriam iguais.

```
aws-cloudhsm> findAllKeys 6 1
Keys on server 0(10.0.0.1):
Number of keys found 3
number of keys matched from start index 0::3
8(s),9(s),11,17(s)
Key Hash:
```

```
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 0(10.0.0.1)
Keys on server 1(10.0.0.2):
Number of keys found 3
number of keys matched from start index 0::3
8(s),9(s),11(o),17(s)
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 1(10.0.0.2)
```

Esse comando demonstra que o valor do hash representa os dados do usuário para todas as chaves no HSM. O comando usa `findAllKeys` para o usuário 3. Ao contrário do usuário 6, que possui ou compartilha apenas 3 chaves, o usuário 3 possui ou compartilha 17 chaves, mas o valor de hash da chave é o mesmo.

```
aws-cloudhsm> findAllKeys 3 1
Keys on server 0(10.0.0.1):
Number of keys found 17
number of keys matched from start index 0::17
6(o),7(o),8(s),11(o),12(o),14(o),262159(o),262160(o),17(s),262162(s),19(s),20(o),21(o),262177(o)
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 0(10.0.0.1)
Keys on server 1(10.0.0.2):
Number of keys found 17
number of keys matched from start index 0::17
6(o),7(o),8(s),11(o),12(o),14(o),262159(o),262160(o),17(s),262162(s),19(s),20(o),21(o),262177(o)
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 1(10.0.0.2)
```

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
findAllKeys <user id> <key hash (0/1)> [<output file>]
```

<user id>

Obtém todas as chaves que o usuário especificado possui ou compartilha. Insira a ID de usuário de um usuário no HSMs. Para encontrar o usuário IDs de todos os usuários, use [ListUsers](#).

Todos os usuários IDs são válidos, mas `findAllKeys` retornam as chaves somente para usuários criptográficos (CUs).

Obrigatório: Sim

<key hash>

Inclui (1) ou exclui (0) um hash da propriedade do usuário e dos dados de compartilhamento para todas as chaves em cada HSM.

Quando o argumento `user id` representa um usuário que possui ou compartilha chaves, o hash da chave é preenchido. O valor do hash da chave é idêntico para todos os usuários que possuem ou compartilham chaves no HSM, mesmo possuindo e compartilhando chaves diferentes. No entanto, quando `user id` representa um usuário que não possui ou compartilha nenhuma chave, como um CO, o valor do hash não é preenchido.

Obrigatório: Sim

<output file>

Grava a saída no arquivo especificado.

Obrigatório: não

Padrão: Stdout

Tópicos relacionados

- [changePswd](#)
- [deleteUser](#)
- [listUsers](#)
- [syncUser](#)
- [findKey](#) em `key_mgmt_util`
- [getKeyInfo](#) em `key_mgmt_util`

Obtenha um valor de atributo AWS CloudHSM chave usando CMU

Use o `getAttribute` comando no AWS CloudHSM `cloudhsm_mgmt_util` (CMU) para obter um valor de atributo para uma chave de todos os módulos de segurança de hardware (HSM) no AWS CloudHSM cluster e gravá-lo em `stdout` (saída padrão) ou em um arquivo. Somente usuários criptográficos (CUs) podem executar esse comando.

Atributos de chave são propriedades de uma chave. Eles incluem características, como o tipo de chave, a classe, o rótulo e o ID, e valores que representam ações que você pode realizar na chave, como criptografar, descriptografar, encapsular, assinar e verificar.

Você só pode usar `getAttribute` nas chaves que você possui e em chaves que são compartilhadas com você. Você pode executar esse comando ou o comando [getAttribute](#) na `key_mgmt_util`, que grava um ou todos os valores de atributos de uma chave em um arquivo.

Para obter uma lista de atributos e das constantes que os representam, use o comando [listAttributes](#). Para alterar os valores de atributo de chaves existentes, use [setAttribute](#) na `key_mgmt_util` e [setAttribute](#) na `cloudhsm_mgmt_util`. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Usuários de criptografia (CU)

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
getAttribute <key handle> <attribute id> [<filename>]
```

Exemplo

Este exemplo obtém o valor do atributo extraível de uma chave no HSMs. Você pode usar um comando como esse para determinar se é possível exportar uma chave do HSMs.

O primeiro comando usa [listAttributes](#) para encontrar a constante que representa o atributo extraível. A saída mostra que a constante de OBJ_ATTR_EXTRACTABLE é 354. Também é possível encontrar essas informações com as descrições dos atributos e seus valores em [AWS CloudHSM referência de atributo-chave para KMU](#).

```
aws-cloudhsm> listAttributes
```

Following are the possible attribute values for getAttribute:

OBJ_ATTR_CLASS	= 0
OBJ_ATTR_TOKEN	= 1
OBJ_ATTR_PRIVATE	= 2
OBJ_ATTR_LABEL	= 3
OBJ_ATTR_TRUSTED	= 134
OBJ_ATTR_KEY_TYPE	= 256
OBJ_ATTR_ID	= 258
OBJ_ATTR_SENSITIVE	= 259
OBJ_ATTR_ENCRYPT	= 260
OBJ_ATTR_DECRYPT	= 261
OBJ_ATTR_WRAP	= 262
OBJ_ATTR_UNWRAP	= 263
OBJ_ATTR_SIGN	= 264
OBJ_ATTR_VERIFY	= 266
OBJ_ATTR_DERIVE	= 268
OBJ_ATTR_LOCAL	= 355
OBJ_ATTR_MODULUS	= 288
OBJ_ATTR_MODULUS_BITS	= 289
OBJ_ATTR_PUBLIC_EXPONENT	= 290
OBJ_ATTR_VALUE_LEN	= 353
OBJ_ATTR_EXTRACTABLE	= 354
OBJ_ATTR_NEVER_EXTRACTABLE	= 356
OBJ_ATTR_ALWAYS_SENSITIVE	= 357
OBJ_ATTR_DESTROYABLE	= 370
OBJ_ATTR_KCV	= 371
OBJ_ATTR_WRAP_WITH_TRUSTED	= 528
OBJ_ATTR_WRAP_TEMPLATE	= 1073742353
OBJ_ATTR_UNWRAP_TEMPLATE	= 1073742354
OBJ_ATTR_ALL	= 512

O segundo comando é usado `getAttribute` para obter o valor do atributo extraível da chave com o identificador de chave 262170 no HSMs. Para especificar o atributo extraível, o comando usa 354, a constante que representa o atributo. Como o comando não especifica um nome de arquivo, `getAttribute` grava a saída em `stdout`.

A saída mostra que o valor do atributo extraível é 1 em todos os HSMs. Esse valor indica que o proprietário da chave pode exportá-la. Quando o valor é 0 (0x0), ele não pode ser exportado do. HSMs Você define o valor do atributo extraível ao criar uma chave, mas não pode alterá-la.

```
aws-cloudhsm> getAttribute 262170 354

Attribute Value on server 0(10.0.1.10):
OBJ_ATTR_EXTRACTABLE
0x00000001

Attribute Value on server 1(10.0.1.12):
OBJ_ATTR_EXTRACTABLE
0x00000001

Attribute Value on server 2(10.0.1.7):
OBJ_ATTR_EXTRACTABLE
0x00000001
```

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
getAttribute <key handle> <attribute id> [<filename>]
```

<key-handle>

Especifica o identificador da chave de destino. Você pode especificar apenas uma chave em cada comando. Para obter o identificador de chave de uma chave, use [findKey](#) na `key_mgmt_util`.

Você deve ter a chave especificada ou ela deve ser compartilhada com você. Para encontrar os usuários de uma chave, use [getKeyInfo](#) em `key_mgmt_util`.

Obrigatório: Sim

<attribute id>

Identifica o atributo. Insira uma constante que represente um atributo, ou 512, que represente todos os atributos. Por exemplo, para obter o tipo de chave, insira 256, que é a constante para o atributo OBJ_ATTR_KEY_TYPE.

Para listar os atributos e suas constantes, use [listAttributes](#). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Obrigatório: Sim

<filename>

Grava a saída no arquivo especificado. Insira um caminho de arquivo.

Se o arquivo especificado existir, getAttribute substitui o arquivo sem aviso prévio.

Obrigatório: não

Padrão: Stdout

Tópicos relacionados

- [getAttribute](#) em key_mgmt_util
- [listAttributes](#)
- [setAttribute](#) em cloudhsm_mgmt_util
- [setAttribute](#) em key_mgmt_util
- [Referência de atributos de chave](#)

Obtenha informações de hardware para cada HSM em um AWS CloudHSM cluster com CMU

Use o getHSMInfo comando no AWS CloudHSM cloudhsm_mgmt_util (CMU) para obter informações sobre o hardware no qual cada módulo de segurança de hardware (HSM) é executado, incluindo o modelo, número de série, estado FIPS, memória, temperatura e os números de versão do hardware e do firmware. As informações também incluem o ID de servidor que a cloudhsm_mgmt_util usa para fazer referência ao HSM.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

Este comando não possui parâmetros.

```
getHSMInfo
```

Exemplo

Este exemplo é usado `getHSMInfo` para obter informações sobre o HSMs no cluster.

```
aws-cloudhsm> getHSMInfo
Getting HSM Info on 3 nodes
*** Server 0 HSM Info ***

Label           :cavium
Model           :NITROX-III CNN35XX-NFBE

Serial Number   :3.0A0101-ICM000001
HSM Flags       :0
FIPS state      :2 [FIPS mode with single factor authentication]

Manufacturer ID :
Device ID       :10
Class Code      :100000
System vendor ID :177D
SubSystem ID    :10
```

```

TotalPublicMemory      :560596
FreePublicMemory       :294568
TotalPrivateMemory     :0
FreePrivateMemory      :0

Hardware Major         :3
Hardware Minor         :0

Firmware Major         :2
Firmware Minor         :03

Temperature            :56 C

Build Number           :13

Firmware ID            :xxxxxxxxxxxxxxxx

```

...

Tópicos relacionados

- [info](#)

Obtenha informações AWS CloudHSM do usuário sobre uma chave usando CMU

Use o `getKeyInfo` comando no AWS CloudHSM `key_mgmt_util` (KMU) para retornar o usuário do módulo de segurança de hardware (HSM) IDs dos usuários que podem usar a chave, incluindo o proprietário e os usuários criptográficos (CU) com os quais a chave é compartilhada. Quando a autenticação de quorum está habilitada em uma chave, `getKeyInfo` também retorna o número de usuários que devem aprovar as operações criptográficas que usam essa chave. Você pode executar `getKeyInfo` somente nas chaves que possui e em chaves que são compartilhadas com você.

Quando você executa `getKeyInfo` em chaves públicas, `getKeyInfo` retorna apenas o proprietário da chave, mesmo que todos os usuários do HSM possam usar a chave pública. Para encontrar o usuário HSM IDs dos usuários em seu HSMs, use [ListUsers](#). Para encontrar as chaves de um usuário específico, use [findKey](#) -u na `key_mgmt_util`. Os agentes de criptografia podem usar [findAllKeys](#) em `cloudhsm_mgmt_util`.

Você possui as chaves que cria. Você pode compartilhar uma chave com outros usuários ao criá-la. Em seguida, para compartilhar ou descompartilhar uma chave existente, use [shareKey](#) em `cloudhsm_mgmt_util`.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CU)

Sintaxe

```
getKeyInfo -k <key-handle> [<output file>]
```

Exemplos

Esses exemplos mostram como usar getKeyInfo para obter informações sobre os usuários de uma chave.

Example: Obter os usuários para uma chave assimétrica

Esse comando obtém os usuários que podem usar a chave AES (assimétrica) com o identificador de chave 262162. A saída mostra que o usuário 3 possui a chave e a compartilhou com os usuários 4 e 6.

Somente os usuários 3, 4 e 6 podem executar getKeyInfo na chave 262162.

```
aws-cloudhsm>getKeyInfo 262162
Key Info on server 0(10.0.0.1):

    Token/Flash Key,

    Owned by user 3

    also, shared to following 2 user(s):

        4
        6
```

```
Key Info on server 1(10.0.0.2):

    Token/Flash Key,

    Owned by user 3

    also, shared to following 2 user(s):

        4
        6
```

Example: Obter usuários para um par de chaves simétricas

Esses comandos usam `getKeyInfo` para obter os usuários que podem usar as chaves em um [par de chaves ECC \(simétrico\)](#). A chave pública possui o identificador de chave 262179. A chave privada possui o identificador de chave 262177.

Quando você `getKeyInfo` executa a chave privada (262177), ela retorna o proprietário da chave (3) e os usuários criptográficos (CUs) 4, com os quais a chave é compartilhada.

```
aws-cloudhsm>getKeyInfo -k 262177
Key Info on server 0(10.0.0.1):

    Token/Flash Key,

    Owned by user 3

    also, shared to following 1 user(s):

        4
Key Info on server 1(10.0.0.2):

    Token/Flash Key,

    Owned by user 3

    also, shared to following 1 user(s):

        4
```

Quando você executa `getKeyInfo` na chave pública (262179), ele retorna apenas o proprietário da chave, o usuário 3.

```
aws-cloudhsm>getKeyInfo -k 262179
```

```
Key Info on server 0(10.0.3.10):
```

```
Token/Flash Key,
```

```
Owned by user 3
```

```
Key Info on server 1(10.0.3.6):
```

```
Token/Flash Key,
```

```
Owned by user 3
```

Para confirmar que o usuário 4 pode usar a chave pública (e todas as chaves públicas no HSM), use o parâmetro `-u` de [findKey](#) em `key_mgmt_util`.

A saída mostra que o usuário 4 pode usar tanto a chave pública (262179) quanto a privada (262177) no par de chaves. O Usuário 4 também pode usar todas as outras chaves públicas e quaisquer chaves privadas que ele tenha criado ou que tenha sido compartilhada com ele.

```
Command: findKey -u 4
```

```
Total number of keys present 8
```

```
number of keys matched from start index 0::7
```

```
11, 12, 262159, 262161, 262162, 19, 20, 21, 262177, 262179
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: Obter o valor de autenticação de quorum (`m_value`) para uma chave

Este exemplo mostra como obter o `m_value` para uma chave. O `m_value` é o número de usuários no quorum que deve aprovar todas as operações criptográficas que usam a chave e as operações para compartilhar e descompartilhar a chave.

Quando a autenticação de quorum está habilitada em uma chave, um quorum de usuários deve aprovar todas as operações criptográficas que usam essa chave. Para habilitar a autenticação de quorum e definir o tamanho do quorum, use o parâmetro `-m_value` ao criar a chave.

Esse comando é usado [genSymKey](#) para criar uma chave AES de 256 bits que é compartilhada com o usuário 4. Ele usa o parâmetro `m_value` para habilitar a autenticação de quorum e definir o tamanho do quorum para dois usuários. O número de usuários deve ser grande o suficiente para fornecer as aprovações necessárias.

A saída mostra que o comando criou a chave 10.

```
Command:  genSymKey -t 31 -s 32 -l aes256m2 -u 4 -m_value 2

Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Created.  Key Handle: 10

Cluster Error Status
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Este comando usa `getKeyInfo` na `cloudhsm_mgmt_util` para obter informações sobre os usuários da chave 10. A saída mostra que a chave é de propriedade do usuário 3 e compartilhada com o usuário 4. Também mostra que um quorum de dois usuários deve aprovar todas as operações criptográficas que usam a chave.

```
aws-cloudhsm>getKeyInfo 10

Key Info on server 0(10.0.0.1):

Token/Flash Key,

Owned by user 3

also, shared to following 1 user(s):

    4
    2 Users need to approve to use/manage this key
Key Info on server 1(10.0.0.2):

Token/Flash Key,

Owned by user 3

also, shared to following 1 user(s):
```


4

2 Users need to approve to use/manage this key

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
getKeyInfo -k <key-handle> <output file>
```

<key-handle>

Especifica o identificador da chave no HSM. Insira o identificador de uma chave que você possui ou compartilha. Esse parâmetro é obrigatório.

Obrigatório: Sim

<output file>

Grava a saída no arquivo especificado, em vez de em stdout. Se o arquivo existir, o comando o substituirá sem aviso prévio.

Obrigatório: não

Padrão: stdout

Tópicos relacionados

- [getKeyInfo](#) em key_mgmt_util
- [findKey](#) em key_mgmt_util
- [findAllKey](#) em cloudhsm_mgmt_util
- [listUsers](#)
- [shareKey](#)

Obtenha informações para cada HSM em um AWS CloudHSM cluster usando CMU

Use o info comando no AWS CloudHSM cloudhsm_mgmt_util (CMU) para obter informações sobre cada um dos módulos de segurança de hardware (HSM) no AWS CloudHSM cluster, incluindo

o nome do host, a porta, o endereço IP e o nome e o tipo do usuário que está conectado ao cloudhsm_mgmt_util no HSM.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
info server <server ID>
```

Exemplo

Este exemplo usa info para obter informações sobre um HSM no cluster. O comando usa 0 para se referir ao primeiro HSM do cluster. A saída mostra o endereço IP, a porta e o tipo e nome do usuário atual.

```
aws-cloudhsm> info server 0
```

Id	Name	Hostname	Port	State	Partition
	LoginState				
0	10.0.0.1	10.0.0.1	2225	Connected	hsm-udw0tkfg1ab
	Logged in as 'testuser(CU)'				

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
info server <server ID>
```

<server id>

Especifica o ID do servidor do HSM. HSMs São atribuídos números ordinais que representam a ordem em que são adicionados ao cluster, começando com 0. Para encontrar o ID do servidor de um HSM, use `getHSMInfo`.

Obrigatório: Sim

Tópicos relacionados

- [getHSMInfo](#)
- [loginHSM e logoutHSM](#)

Listar os atributos de uma AWS CloudHSM chave usando CMU

Use o `listAttributes` comando no AWS CloudHSM `cloudhsm_mgmt_util` (CMU) para listar os atributos de uma chave e as constantes que os representam. AWS CloudHSM Você usa essas constantes para identificar os atributos nos comandos [getAttribute](#) e [setAttribute](#).

Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

```
listAttributes [-h]
```

Exemplo

Este comando lista os principais atributos que você pode obter e alterar em `key_mgmt_util`, bem como as constantes que os representam. Para obter ajuda sobre a interpretação dos principais

atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#). Para representar todos os atributos, use 512.

Command: **listAttributes**

Description

=====

The following are all of the possible attribute values for getAttribute.

OBJ_ATTR_CLASS	= 0
OBJ_ATTR_TOKEN	= 1
OBJ_ATTR_PRIVATE	= 2
OBJ_ATTR_LABEL	= 3
OBJ_ATTR_TRUSTED	= 134
OBJ_ATTR_KEY_TYPE	= 256
OBJ_ATTR_ID	= 258
OBJ_ATTR_SENSITIVE	= 259
OBJ_ATTR_ENCRYPT	= 260
OBJ_ATTR_DECRYPT	= 261
OBJ_ATTR_WRAP	= 262
OBJ_ATTR_UNWRAP	= 263
OBJ_ATTR_SIGN	= 264
OBJ_ATTR_VERIFY	= 266
OBJ_ATTR_DERIVE	= 268
OBJ_ATTR_LOCAL	= 355
OBJ_ATTR_MODULUS	= 288
OBJ_ATTR_MODULUS_BITS	= 289
OBJ_ATTR_PUBLIC_EXPONENT	= 290
OBJ_ATTR_VALUE_LEN	= 353
OBJ_ATTR_EXTRACTABLE	= 354
OBJ_ATTR_NEVER_EXTRACTABLE	= 356
OBJ_ATTR_ALWAYS_SENSITIVE	= 357
OBJ_ATTR_DESTROYABLE	= 370
OBJ_ATTR_KCV	= 371
OBJ_ATTR_WRAP_WITH_TRUSTED	= 528
OBJ_ATTR_WRAP_TEMPLATE	= 1073742353
OBJ_ATTR_UNWRAP_TEMPLATE	= 1073742354
OBJ_ATTR_ALL	= 512

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

Tópicos relacionados

- [getAttribute](#)
- [setAttribute](#)
- [Referência de atributos de chave](#)

Listar todos os AWS CloudHSM usuários usando CMU

Use o listUsers comando no AWS CloudHSM cloudhsm_mgmt_util para colocar os usuários em cada um dos módulos de segurança de hardware (HSM), junto com o tipo de usuário e outros atributos. Todos os tipos de usuários podem executar esse comando. Você nem mesmo precisa estar conectado à cloudhsm_mgmt_util para executar esse comando.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

Este comando não possui parâmetros.

```
listUsers
```

Exemplo

Esse comando lista os usuários em cada um dos HSMs no cluster e exibe seus atributos. Você pode usar o atributo `User ID` para identificar usuários em outros comandos, como `deleteUser`, `changePswd` e `findAllKeys`.

```
aws-cloudhsm> listUsers
```

```
Users on server 0(10.0.0.1):
```

```
Number of users found:6
```

User Id	User Type	User Name	MofnPubKey	
LoginFailureCnt	2FA			
1	PC0	admin	YES	0
NO				
2	AU	app_user	NO	0
NO				
3	CU	crypto_user1	NO	0
NO				
4	CU	crypto_user2	NO	0
NO				
5	C0	officer1	YES	0
NO				
6	C0	officer2	NO	0
NO				

```
Users on server 1(10.0.0.2):
```

```
Number of users found:5
```

User Id	User Type	User Name	MofnPubKey	
LoginFailureCnt	2FA			
1	PC0	admin	YES	0
NO				
2	AU	app_user	NO	0
NO				
3	CU	crypto_user1	NO	0
NO				
4	CU	crypto_user2	NO	0
NO				
5	C0	officer1	YES	0
NO				

A saída inclui os seguintes atributos do usuário:

- **User ID:** identifica o usuário em comandos da `key_mgmt_util` e do [cloudhsm_mgmt_util](#).

- [User type](#): determina as operações que o usuário pode executar no HSM.
- User Name: exibe o nome amigável definido pelo usuário para o usuário.
- MofnPubKey: indica se o usuário registrou um par de chaves para assinar tokens de [autenticação de quórum](#).
- LoginFailureCnt: indica o número de vezes que o usuário fez login sem sucesso.
- 2FA: indica que o usuário habilitou a autenticação multifator.

Tópicos relacionados

- [listUsers](#) em key_mgmt_util
- [createUser](#)
- [deleteUser](#)
- [changePswd](#)

Faça login e saia de um HSM usando o AWS CloudHSM Management Utility

Você pode usar os comandos loginHSM e logoutHSM no cloudhsm_mgmt_util do AWS CloudHSM para fazer login e logout de cada HSM em um cluster. Qualquer usuário de qualquer tipo pode usar esses comandos.

Note

Se você exceder cinco tentativas incorretas de login, sua conta será bloqueada. Para desbloquear a conta, um responsável pela criptografia (CO) precisará redefinir sua senha usando o comando [changePswd](#) no cloudhsm_mgmt_util.

Para solucionar problemas de loginHSM e logoutHSM

Antes de executar estes comandos cloudhsm_mgmt_util, você deve iniciar cloudhsm_mgmt_util.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração que o AWS CloudHSM cliente e as ferramentas de linha de comando usam. Caso contrário, as alterações feitas podem não ser efetivas HSMs em todas as partes do cluster.

Se você tiver mais de um HSM no cluster, pode ser que mais tentativas de login sejam permitidas antes de a sua conta ser bloqueada. Isso ocorre porque o cliente CloudHSM equilibra a carga em

vários. HSMs Sendo assim, pode ser que as tentativas de login não comecem sempre no mesmo HSM. Se você estiver testando essa funcionalidade, recomendamos fazer isso em um cluster com apenas um HSM ativo.

Se você criou seu cluster antes de fevereiro de 2018, sua conta será bloqueada após 20 tentativas incorretas de login.

Tipo de usuário

Os usuários a seguir podem executar esses comandos.

- Responsável pela pré-criptografia (PRECO)
- Responsável pela criptografia (CO)
- Crypto user (CU) (usuário de criptografia)

Sintaxe

Insira os argumentos na ordem especificada no diagrama de sintaxe. Use o parâmetro `-hpswd` para mascarar sua senha. Para fazer login com autenticação de dois fatores (2FA), use o parâmetro `-2fa` e inclua um caminho de arquivo. Para obter mais informações, consulte [the section called “Argumentos”](#).

```
loginHSM <user-type> <user-name> <password> | -hpswd> [-2fa </path/to/authdata>]
```

```
logoutHSM
```

Exemplos

Esses exemplos mostram como usar `loginHSM` e `logoutHSM` fazer login e sair de tudo HSMs em um cluster.

Example: faça login no HSMs em um cluster

Esse comando conecta você a todos HSMs em um cluster com as credenciais de um usuário CO chamado `admin` e uma senha `deco12345`. A saída mostra que o comando foi bem-sucedido e que você se conectou ao HSMs (que, nesse caso, é `server 0` e `server 1`).

```
aws-cloudhsm>loginHSM CO admin deco12345
```



```
loginHSM success on server 0(10.0.2.9)
loginHSM success on server 1(10.0.3.11)
```

Example: Faça login com uma senha oculta

Esse comando é o mesmo do exemplo acima, exceto que, desta vez, você especifica que o sistema deve ocultar a senha.

```
aws-cloudhsm>loginHSM C0 admin -hpswd
```

O sistema solicita que você forneça sua senha. Você insere a senha, o sistema oculta a senha e a saída mostra que o comando foi bem-sucedido e que você se conectou ao HSMs.

```
Enter password:

loginHSM success on server 0(10.0.2.9)
loginHSM success on server 1(10.0.3.11)

aws-cloudhsm>
```

Example: fazer logout de um HSM

Esse comando desconecta você do HSMs que você está conectado no momento (que, nesse caso, é server 0 e server 1). A saída mostra que o comando foi bem-sucedido e que você se desconectou do HSMs.

```
aws-cloudhsm>logoutHSM

logoutHSM success on server 0(10.0.2.9)
logoutHSM success on server 1(10.0.3.11)
```

Argumentos

Insira os argumentos na ordem especificada no diagrama de sintaxe. Use o parâmetro `-hpswd` para mascarar sua senha. Para fazer login com autenticação de dois fatores (2FA), use o parâmetro `-2fa` e inclua um caminho de arquivo. Para obter mais informações sobre como trabalhar com 2FA, consulte [Gerenciar a 2FA para usuários](#)

```
loginHSM <user-type> <user-name> <password> |-hpswd> [-2fa </path/to/authdata>]
```

<user type>

Especifica o tipo de usuário que está fazendo login no HSMs. Para obter mais informações, consulte [Tipo de usuário](#) acima.

Obrigatório: Sim

<user name>

Especifica o nome de usuário do usuário que está fazendo login no HSMs.

Obrigatório: Sim

<password | -hpswd >

Especifica a senha do usuário que está fazendo login no HSMs. Para ocultar sua senha, use o parâmetro -hpswd no lugar da senha e siga as instruções.

Obrigatório: Sim

[-2fa </ >path/to/authdata]

Especifica que o sistema deve usar um segundo fator para autenticar esse usuário de CO habilitado para 2FA. Para obter os dados necessários para fazer login com a 2FA, inclua um caminho para um local no sistema de arquivos com um nome de arquivo após o parâmetro -2fa. Para obter mais informações sobre como trabalhar com 2FA, consulte [Gerenciar a 2FA para usuários](#).

Obrigatório: não

Tópicos relacionados

- [Conceitos básicos de cloudhsm_mgmt_util](#)
- [Ativar o cluster](#)

Associe AWS CloudHSM usuários a chaves usando CMU

Use o registerQuorumPubKey comando no AWS CloudHSM cloudhsm_mgmt_util para associar usuários do módulo de segurança de hardware (HSM) a pares de chaves RSA-2048 assimétricos. Depois de associar usuários do HSM às chaves, esses usuários podem usar a chave privada para aprovar solicitações de quórum e o cluster pode usar a chave pública registrada para verificar se

a assinatura pertence ao usuário. Para obter mais informações sobre a autenticação de quórum, consulte [Gerenciar a autenticação de quórum \(controle de acesso M of N\)](#).

Tip

Na AWS CloudHSM documentação, a autenticação de quórum às vezes é chamada de M de N (MoFN), o que significa um mínimo de M aprovadores de um número total de N aprovadores.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Responsáveis pela criptografia (CO)

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
registerQuorumPubKey <user-type> <user-name> <registration-token> <signed-registration-token> <public-key>
```

Exemplos

Este exemplo mostra como usar `registerQuorumPubKey` para registrar agentes criptográficos (CO) como aprovadores em solicitações de autenticação de quórum. Para executar esse comando, você deve ter um par de chaves RSA-2048 assimétrico, um token assinado e um token não assinado. Para receber mais informações sobre os requisitos, consulte [the section called “Argumentos”](#).

Example: Registrar um usuário do HSM para autenticação de quórum

Este exemplo registra um CO nomeado `quorum_officer` como aprovador da autenticação de quórum.

```
aws-cloudhsm> registerQuorumPubKey CO <quorum_officer> </path/to/  
quorum_officer.token> </path/to/quorum_officer.token.sig> </path/to/quorum_officer.pub>
```

```
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?y
registerQuorumPubKey success on server 0(10.0.0.1)
```

O comando final usa o comando [listUsers](#) para verificar se quorum_officer está registrado como um usuário MoFN.

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:3
```

User Id	User Type	User Name	MofnPubKey
LoginFailureCnt	2FA		
1	PC0	admin	N0
0	N0		
2	AU	app_user	N0
0	N0		
3	C0	quorum_officer	YES
0	N0		

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
registerQuorumPubKey <user-type> <user-name> <registration-token> <signed-registration-token> <public-key>
```

<user-type>

Especifica o tipo de usuário. Esse parâmetro é obrigatório.

Para obter informações detalhadas sobre os tipos de usuários em um HSM, consulte [Tipos de usuário do HSM para o AWS CloudHSM Management Utility](#).

Valores válidos:

- CO: Oficiais de criptografia podem gerenciar usuários, mas não conseguem gerenciar chaves.

Obrigatório: Sim

<user-name>

Especifica um nome amigável para o usuário. O tamanho máximo é de 31 caracteres. O único caractere especial permitido é um sublinhado (_).

Não é possível alterar o nome de um usuário após sua criação. Em comandos da `cloudhsm_mgmt_util`, o tipo de usuário e a senha diferenciam maiúsculas de minúsculas, mas o nome do usuário não diferencia.

Obrigatório: Sim

<registration-token>

Especifica o caminho para um arquivo que contém um token de registro não assinado. Pode ter qualquer dado aleatório com tamanho máximo de arquivo de 245 bytes. Para obter mais informações sobre como criar um token de registro não assinado, consulte [Criar e assinar um token de registro](#).

Obrigatório: Sim

<signed-registration-token>

Especifica o caminho para um arquivo que contém o hash assinado pelo mecanismo SHA256 _PKCS do token de registro. Para obter mais informações, consulte [Criar e assinar um token de registro](#).

Obrigatório: Sim

<public-key>

Especifica o caminho para o arquivo que contém uma chave pública de um par de chaves assimétrico RSA-2048. Use a chave privada para assinar o token de registro. Para obter mais informações, consulte [Criar um par de chaves](#).

Obrigatório: Sim

 Note

O cluster usa a mesma chave para autenticação de quórum e para autenticação de dois fatores (2FA). Isso significa que você não pode alternar uma chave de quórum para

um usuário que tenha o 2FA ativado usando `registerQuorumPubKey`. Para alternar a chave, você deve usar `changePswd`. Para obter mais informações sobre como usar a autenticação de quórum e a 2FA, consulte [Autenticação de quórum e 2FA](#).

Tópicos relacionados

- [Criar um par de chaves RSA](#)
- [Crie e assine um token de registro](#)
- [Registrar uma chave pública no HSM](#)
- [Aplicar autenticação de quorum \(controle de acesso M de N\)](#)
- [Autenticação de quórum e 2FA](#)
- [listUsers](#)

Interaja com um HSM em um AWS CloudHSM cluster usando CMU

Use o server comando no AWS CloudHSM `cloudhsm_mgmt_util` para entrar no modo servidor e interagir diretamente com uma instância específica do módulo de segurança de hardware (HSM).

Normalmente, quando você emite um comando em `cloudhsm_mgmt_util`, o comando afeta tudo HSMs no cluster designado (modo global). No entanto, pode haver circunstâncias para as quais você precisa emitir comandos para um único HSM. Por exemplo, se a sincronização automática falhar, poderá ser necessário sincronizar chaves e usuários em um HSM para manter a consistência em todo o cluster.

Após o início bem-sucedido, o prompt de comando `aws-cloudhsm >` será substituído pelo prompt de comando `server >`.

Para usar sair do modo de servidor, use o comando `exit`. Após a saída bem-sucedida, você será redirecionado ao prompt de comando da `cloudhsm_mgmt_util`.

Antes de executar os comandos `cloudhsm_mgmt_util`, você deve iniciar `cloudhsm_mgmt_util`.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Todos os usuários.

Pré-requisitos

Para entrar no modo de servidor, será necessário primeiro saber o número do servidor do HSM de destino. Os números do servidor são listados na saída de rastreamento gerada pela `cloudhsm_mgmt_util` após ser iniciada. Os números do servidor são atribuídos na mesma ordem em que HSMs aparecem no arquivo de configuração. Para este exemplo, supomos que `server 0` seja o servidor que corresponde ao HSM desejado.

Sintaxe

Para iniciar o modo de servidor:

```
server <server-number>
```

Para sair do modo de servidor:

```
server> exit
```

Exemplo

Esse comando insere o modo de servidor em um HSM com o servidor número 0.

```
aws-cloudhsm> server 0  
  
Server is in 'E2' mode...
```

Para usar sair do modo de servidor, use o comando `exit`.

```
server0> exit
```

Argumentos

```
server <server-number>
```

<server-number>

Especifica o número do servidor do HSM de destino.

Obrigatório: Sim

Não há argumentos para o comando `exit`.

Tópicos relacionados

- [syncKey](#)
- [createUser](#)
- [deleteUser](#)

Defina os atributos das AWS CloudHSM chaves usando CMU

Use o `setAttribute` comando no AWS CloudHSM `cloudhsm_mgmt_util` para alterar o valor do rótulo, criptografar, descriptografar, encapsular e desempacotar os atributos de uma chave no. HSMs Você também pode usar o comando [setAttribute](#) na `key_mgmt_util` para converter uma chave de sessão em uma chave persistente. Você só pode alterar os atributos das chaves que possui.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Usuários de criptografia (CU)

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
setAttribute <key handle> <attribute id>
```

Exemplo

Esse exemplo mostra como desabilitar a funcionalidade de descriptografia de uma chave simétrica. É possível usar um comando como esse para configurar uma chave de encapsulamento, que deve

ser capaz de encapsular e desencapsular outras chaves, mas não de criptografar ou descriptografar dados.

A primeira etapa é criar a chave de encapsulamento. Esse comando é usado [genSymKey](#) em `key_mgmt_util` para gerar uma chave simétrica AES de 256 bits. A saída mostra que a nova chave tem o identificador de chave 14.

```
$ genSymKey -t 31 -s 32 -l aes256
```

```
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Created. Key Handle: 14
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Em seguida, queremos confirmar o valor atual do atributo de descriptografia. Para obter o ID do atributo de descriptografia, use [listAttributes](#). A saída mostra que a constante que representa o atributo `OBJ_ATTR_DECRYPT` é 261. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

```
aws-cloudhsm> listAttributes
```

```
Following are the possible attribute values for getAttribute:
```

<code>OBJ_ATTR_CLASS</code>	= 0
<code>OBJ_ATTR_TOKEN</code>	= 1
<code>OBJ_ATTR_PRIVATE</code>	= 2
<code>OBJ_ATTR_LABEL</code>	= 3
<code>OBJ_ATTR_TRUSTED</code>	= 134
<code>OBJ_ATTR_KEY_TYPE</code>	= 256
<code>OBJ_ATTR_ID</code>	= 258
<code>OBJ_ATTR_SENSITIVE</code>	= 259
<code>OBJ_ATTR_ENCRYPT</code>	= 260
<code>OBJ_ATTR_DECRYPT</code>	= 261
<code>OBJ_ATTR_WRAP</code>	= 262
<code>OBJ_ATTR_UNWRAP</code>	= 263
<code>OBJ_ATTR_SIGN</code>	= 264
<code>OBJ_ATTR_VERIFY</code>	= 266
<code>OBJ_ATTR_DERIVE</code>	= 268
<code>OBJ_ATTR_LOCAL</code>	= 355
<code>OBJ_ATTR_MODULUS</code>	= 288

```

OBJ_ATTR_MODULUS_BITS           = 289
OBJ_ATTR_PUBLIC_EXPONENT        = 290
OBJ_ATTR_VALUE_LEN              = 353
OBJ_ATTR_EXTRACTABLE            = 354
OBJ_ATTR_NEVER_EXTRACTABLE      = 356
OBJ_ATTR_ALWAYS_SENSITIVE       = 357
OBJ_ATTR_DESTROYABLE            = 370
OBJ_ATTR_KCV                    = 371
OBJ_ATTR_WRAP_WITH_TRUSTED      = 528
OBJ_ATTR_WRAP_TEMPLATE          = 1073742353
OBJ_ATTR_UNWRAP_TEMPLATE        = 1073742354
OBJ_ATTR_ALL                     = 512

```

Para obter o valor atual do atributo de descryptografia para a chave 14, o próximo comando [getAttribute](#) na `cloudhsm_mgmt_util`.

A saída mostra que o valor do atributo decrypt é verdadeiro (1) em ambos HSMs no cluster.

```

aws-cloudhsm> getAttribute 14 261

Attribute Value on server 0(10.0.0.1):
OBJ_ATTR_DECRYPT
0x00000001

Attribute Value on server 1(10.0.0.2):
OBJ_ATTR_DECRYPT
0x00000001

```

Esse comando usa `setAttribute` para alterar o valor do atributo de descryptografia (atributo 261) da chave 14 para 0. Isso desabilita a funcionalidade de descryptografia na chave.

A saída mostra que o comando foi bem-sucedido em ambos HSMs no cluster.

```

aws-cloudhsm> setAttribute 14 261 0
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

```

```
Do you want to continue(y/n)? y
setAttribute success on server 0(10.0.0.1)
setAttribute success on server 1(10.0.0.2)
```

O comando final repete o comando `getAttribute`. Novamente, ele obtém o atributo decriptografia (atributo 261) da chave 14.

Dessa vez, a saída mostra que o valor do atributo decrypt é false (0) em ambos HSMs no cluster.

```
aws-cloudhsm > getAttribute 14 261
Attribute Value on server 0(10.0.3.6):
OBJ_ATTR_DECRYPT
0x00000000

Attribute Value on server 1(10.0.1.7):
OBJ_ATTR_DECRYPT
0x00000000
```

Argumentos

```
setAttribute <key handle> <attribute id>
```

<key-handle>

Especifica o identificador de uma chave que você possui. Você pode especificar apenas uma chave em cada comando. Para obter o identificador de chave de uma chave, use [findKey](#) na `key_mgmt_util`. Para encontrar os usuários de uma chave, use [getKeyInfo](#).

Obrigatório: Sim

<attribute id>

Especifica a constante que representa o atributo que você deseja alterar. Você pode especificar apenas um atributo em cada comando. Para obter os atributos e seus valores inteiros, use [listAttributes](#). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Valores válidos:

- 3 – OBJ_ATTR_LABEL.
- 134 – OBJ_ATTR_TRUSTED.
- 260 – OBJ_ATTR_ENCRYPT.

- 261 – OBJ_ATTR_DECRYPT.
- 262 – OBJ_ATTR_WRAP.
- 263 – OBJ_ATTR_UNWRAP.
- 264 – OBJ_ATTR_SIGN.
- 266 – OBJ_ATTR_VERIFY.
- 268 – OBJ_ATTR_DERIVE.
- 370 – OBJ_ATTR_DESTROYABLE.
- 528 – OBJ_ATTR_WRAP_WITH_TRUSTED.
- 1073742353 – OBJ_ATTR_WRAP_TEMPLATE.
- 1073742354 – OBJ_ATTR_UNWRAP_TEMPLATE.

Obrigatório: Sim

Tópicos relacionados

- [setAttribute](#) em key_mgmt_util
- [getAttribute](#)
- [listAttributes](#)
- [Referência de atributos de chave](#)

Sair do CMU

Use o quit comando no cloudhsm_mgmt_util para AWS CloudHSM sair do cloudhsm_mgmt_util. Qualquer usuário de qualquer tipo pode usar esse comando.

Antes de executar os comandos cloudhsm_mgmt_util, você deve iniciar cloudhsm_mgmt_util.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Todos os usuários. Você não precisa estar conectado para executar esse comando.

Sintaxe

```
quit
```

Exemplo

Esse comando sai do `cloudhsm_mgmt_util`. Após a conclusão bem-sucedida, você será redirecionado para sua linha de comando normal. Esse comando não possui parâmetros de saída.

```
aws-cloudhsm> quit
```

```
disconnecting from servers, please wait...
```

Tópicos relacionados

- [Conceitos básicos de `cloudhsm\_mgmt\_util`](#)

Compartilhe AWS CloudHSM chaves usando CMU

Use o `shareKey` comando no AWS CloudHSM `cloudhsm_mgmt_util` para compartilhar e cancelar o compartilhamento de chaves que você possui com outros usuários de criptografia. Somente o proprietário da chave pode compartilhá-la e descompartilhá-la. Você também pode compartilhar uma chave ao criá-la.

Os usuários que compartilham a chave podem usá-la em operações criptográficas, mas não podem excluir, exportar, compartilhar ou descompartilhar essa chave, nem alterar seus atributos. Quando a autenticação de quorum está habilitada em uma chave, o quorum deve aprovar todas as operações que compartilham ou descompartilham a chave.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CU)

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

Tipo de usuário: usuário de criptografia (CU)

```
shareKey <key handle> <user id> <(share/unshare key?) 1/0>
```

Exemplo

Os exemplos a seguir mostram como usar shareKey para compartilhar e descompartilhar as chaves que você possui com outros usuários de criptografia.

Example: Compartilhar uma chave

Este exemplo usa shareKey para compartilhar uma [chave privada ECC](#) que o usuário atual possui com outro usuário criptográfico no HSMs. Chaves públicas estão disponíveis para todos os usuários do HSM e, portanto, você não pode compartilhá-las ou descompartilhá-las.

O primeiro comando é usado [getKeyInfo](#) para obter as informações do usuário para a chave 262177, uma chave privada ECC no HSMs.

A saída mostra que a chave 262177 é de propriedade do usuário 3, mas não é compartilhada.

```
aws-cloudhsm>getKeyInfo 262177

Key Info on server 0(10.0.3.10):

    Token/Flash Key,

    Owned by user 3

Key Info on server 1(10.0.3.6):

    Token/Flash Key,

    Owned by user 3
```

Esse comando é usado shareKey para compartilhar a chave 262177 com o usuário 4, outro usuário criptográfico no HSMs. O argumento final usa um valor de 1 para indicar uma operação de compartilhamento.

A saída mostra que a operação foi bem-sucedida em ambos HSMs no cluster.

```
aws-cloudhsm>shareKey 262177 4 1
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?y
shareKey success on server 0(10.0.3.10)
shareKey success on server 1(10.0.3.6)
```

Para verificar se a operação foi bem-sucedida, o exemplo repete o primeiro comando getKeyInfo.

A saída mostra que a chave 262177 agora é compartilhada com o usuário 4.

```
aws-cloudhsm>getKeyInfo 262177

Key Info on server 0(10.0.3.10):

    Token/Flash Key,

    Owned by user 3

    also, shared to following 1 user(s):

        4
Key Info on server 1(10.0.3.6):

    Token/Flash Key,

    Owned by user 3

    also, shared to following 1 user(s):

        4
```

Example: Descompartilhar uma chave

Esse exemplo não compartilha uma chave simétrica, ou seja, ele remove um usuário de criptografia da lista de usuários compartilhados para a chave.

Esse comando usa `shareKey` para remover o usuário 4 da lista de usuários compartilhados para a chave 6. O argumento final usa um valor de 0 para indicar uma operação não compartilhada.

A saída mostra que o comando foi bem-sucedido em ambos HSMs. Como resultado, o usuário 4 não pode mais usar a chave 6 em operações criptográficas.

```
aws-cloudhsm>shareKey 6 4 0
*****CAUTION*****
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
*****

Do you want to continue(y/n)?y
shareKey success on server 0(10.0.3.10)
shareKey success on server 1(10.0.3.6)
```

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
shareKey <key handle> <user id> <(share/unshare key?) 1/0>
```

<key-handle>

Especifica o identificador de uma chave que você possui. Você pode especificar apenas uma chave em cada comando. Para obter o identificador de chave de uma chave, use [findKey](#) na `key_mgmt_util`. Para verificar se você possui uma chave, use [getKeyInfo](#).

Obrigatório: Sim

<user id>

Especifica o ID do usuário de criptografia (CU) com quem você está compartilhando ou descompartilhando a chave. Para encontrar o ID de um usuário, use [listUsers](#).

Obrigatório: Sim

<share 1 or unshare 0>

Para compartilhar a chave com o usuário especificado, digite 1. Para descompartilhar a chave, isto é, para remover o usuário especificado da lista de usuários compartilhados da chave, digite 0.

Obrigatório: Sim

Tópicos relacionados

- [getKeyInfo](#)

Sincronize chaves em todo o AWS CloudHSM cluster usando CMU

Use o `syncKey` comando no AWS CloudHSM `cloudhsm_mgmt_util` para sincronizar manualmente as chaves entre instâncias do HSM em um cluster ou em clusters clonados. Em geral, você não precisa usar esse comando, pois as instâncias do HSM em um cluster sincronizam as chaves automaticamente. No entanto, a sincronização de chaves entre clusters clonados deve ser feita manualmente. Os clusters clonados geralmente são criados em diferentes AWS regiões para simplificar os processos globais de escalabilidade e recuperação de desastres.

Você não pode usar `syncKey` para sincronizar chaves entre clusters arbitrários: um dos clusters deve ter sido criado a partir de um backup do outro. Além disso, para que a operação seja bem-sucedida, os dois clusters devem ter credenciais CO e CU consistentes. Para obter mais informações, consulte [Usuários do HSM](#).

Para usar `syncKey`, primeiro você deve [criar um arquivo de AWS CloudHSM configuração](#) que especifique um HSM do cluster de origem e outro do cluster de destino. Isso permitirá que `cloudhsm_mgmt_util` se conecte às duas instâncias do HSM. Use esse arquivo de configuração para iniciar o `cloudhsm_mgmt_util`. Depois, faça login usando as credenciais de um CO ou de um CU que tenha as chaves que você deseja sincronizar.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Responsáveis pela criptografia (CO)
- Usuários de criptografia (CU)

Note

COs podem ser usados `syncKey` em qualquer tecla, enquanto só CUs podem usar esse comando em teclas que eles possuem. Para obter mais informações, consulte [the section called “Tipos de usuário”](#).

Pré-requisitos

Antes de começar, você deve saber o `key handle` da chave no HSM de origem a ser sincronizada com o HSM de destino. Para encontrar o `key handle`, use o comando [listUsers](#) para relacionar todos os identificadores dos usuários nomeados. Em seguida, use o [findAllKeys](#) comando para encontrar todas as chaves que pertencem a um determinado usuário.

Você também precisa saber o que `server IDs` foi atribuído à origem e ao destino HSMs, que são mostrados na saída de rastreamento retornada pelo `cloudhsm_mgmt_util` na inicialização. Eles são atribuídos na mesma ordem em que HSMs aparecem no arquivo de configuração.

Siga as instruções em [Uso da CMU entre clusters clonados](#) e inicialize a `cloudhsm_mgmt_util` com o novo arquivo de configuração. Depois, entre no modo de servidor no HSM de origem emitindo o comando [server](#).

Sintaxe

Note

Para executar `syncKey`, primeiro entre no modo de servidor no HSM que contém a chave a ser sincronizada.

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

Tipo de usuário: usuário de criptografia (CU)

```
syncKey <key handle> <destination hsm>
```

Exemplo

Execute o comando `server` para fazer login no HSM de origem e digite o modo do servidor. Para este exemplo, estamos supondo que `server 0` é o HSM de origem.

```
aws-cloudhsm> server 0
```

Agora, execute o comando `syncKey`. Neste exemplo, estamos supondo que 261251 deve ser sincronizado com a chave `server 1`.

```
aws-cloudhsm> syncKey 261251 1
syncKey success
```

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
syncKey <key handle> <destination hsm>
```

<key handle>

Especifica o identificador de chave da chave a ser sincronizada. Você pode especificar apenas uma chave em cada comando. Para obter o identificador de uma chave, use [findAllKeys](#) enquanto estiver conectado a um servidor HSM.

Obrigatório: Sim

<destination hsm>

Especifica o número do servidor com o qual você está sincronizando uma chave.

Obrigatório: Sim

Tópicos relacionados

- [listUsers](#)
- [findAllKeys](#)
- [descreve-clusters em](#) AWS CLI
- [servidor](#)

Sincronize usuários em todo o AWS CloudHSM cluster usando CMU

Use o `syncUser` comando no AWS CloudHSM `cloudhsm_mgmt_util` para sincronizar manualmente usuários criptográficos (U) ou agentes criptográficos (CUs) em instâncias de HSM em um cluster ou em clusters COs clonados. AWS CloudHSM não sincroniza automaticamente os usuários. Geralmente, você gerencia usuários no modo global para que todos HSMs em um cluster sejam atualizados juntos. Talvez você precise usar `syncUser` se um HSM for dessincronizado acidentalmente (por exemplo, devido a alterações de senha) ou se você desejar alternar as credenciais de usuário em clusters clonados. Os clusters clonados geralmente são criados em diferentes AWS regiões para simplificar o dimensionamento global e os processos de recuperação de desastres.

Antes de executar qualquer comando da CMU, você deve iniciar a CMU e fazer login no HSM. Certifique-se de fazer login com o tipo de conta de usuário que possa executar os comandos que você planeja usar.

Se você adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações feitas podem não ser efetivas para todos HSMs no cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Responsáveis pela criptografia (CO)

Pré-requisitos

Antes de começar, você deve saber o `user ID` do usuário no HSM de origem a ser sincronizado com o HSM de destino. Para encontrar o `user ID`, use o comando [listUsers](#) para listar todos os usuários do HSMs em um cluster.

Você também precisa saber o que `server ID` foi atribuído à origem e ao destino HSMs, que são mostrados na saída de rastreamento retornada pelo `cloudhsm_mgmt_util` na inicialização. Eles são atribuídos na mesma ordem em que HSMs aparecem no arquivo de configuração.

Se você estiver sincronizando HSMs entre clusters clonados, siga as instruções em [Usando CMU em clusters clonados](#) e inicialize `cloudhsm_mgmt_util` com o novo arquivo de configuração.

Quando estiver pronto para executar `syncUser`, entre no modo de servidor do HSM de origem emitindo o comando [server](#).

Sintaxe

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
syncUser <user ID> <server ID>
```

Exemplo

Execute o comando `server` para fazer login no HSM de origem e entrar no modo de servidor. Para este exemplo, estamos supondo que `server 0` é o HSM de origem.

```
aws-cloudhsm> server 0
```

Agora, execute o comando `syncUser`. Para este exemplo, estamos supondo que o usuário 6 é o usuário a ser sincronizado e que `server 1` é o HSM de destino.

```
server 0> syncUser 6 1
ExtractMaskedObject: 0x0 !
InsertMaskedObject: 0x0 !
syncUser success
```

Argumentos

Como esses comandos não têm parâmetros específicos, insira os argumentos na ordem especificada nos diagramas de sintaxe.

```
syncUser <user ID> <server ID>
```

<user ID>

Especifica o ID do usuário a ser sincronizado. Você pode especificar apenas um usuário em cada comando. Para obter o ID de um usuário, use [listUsers](#).

Obrigatório: Sim

<server ID>

Especifica o número do servidor do HSM com o qual você está sincronizando um usuário.

Obrigatório: Sim

Tópicos relacionados

- [listUsers](#)
- [descreve-clusters em](#) AWS CLI
- [servidor](#)

AWS CloudHSM Utilitário de gerenciamento de chaves (KMU)

O utilitário de gerenciamento de chaves (KMU) é uma ferramenta de linha de comando AWS CloudHSM que ajuda os usuários de criptografia (UC) a gerenciar chaves nos módulos de segurança de hardware (HSM). Ela inclui vários comandos que geram, excluem, importam e exportam chaves, obtêm e definem atributos, localizam chaves e executam operações criptográficas.

O KMU e o CMU fazem parte [do pacote do Client SDK 3](#).

Para começar rapidamente, consulte [Começando com AWS CloudHSM key_mgmt_util](#). Para obter informações detalhadas sobre os comandos, consulte [Referência para comandos do AWS CloudHSM Key Management Utility](#). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Para usar key_mgmt_util se você estiver usando Linux, conecte-se à instância do cliente e, em seguida, consulte [Instale e configure o AWS CloudHSM cliente para KMU \(Linux\)](#). Se você estiver usando Windows, consulte [Instale e configure o AWS CloudHSM cliente para KMU \(Windows\)](#).

Tópicos

- [Começando com AWS CloudHSM key_mgmt_util](#)
- [Instale e configure o AWS CloudHSM cliente para KMU \(Linux\)](#)
- [Instale e configure o AWS CloudHSM cliente para KMU \(Windows\)](#)
- [Referência para comandos do AWS CloudHSM Key Management Utility](#)

Começando com AWS CloudHSM key_mgmt_util

AWS CloudHSM inclui duas ferramentas de linha de comando com o [software AWS CloudHSM cliente](#). A ferramenta [cloudhsm_mgmt_util](#) inclui comandos para gerenciar usuários do HSM. A ferramenta [key_mgmt_util](#) inclui comandos para gerenciar as chaves. Para começar a usar a ferramenta key_mgmt_util da linha de comando, consulte os tópicos a seguir.

Tópicos

- [Configurar AWS CloudHSM key_mgmt_util](#)
- [Faça login no HSMs em um AWS CloudHSM cluster usando KMU](#)
- [Saia do HSMs em um AWS CloudHSM cluster usando KMU](#)
- [Pare o AWS CloudHSM key_mgmt_util](#)

Se você encontrar uma mensagem de erro ou um resultado inesperado para um comando, consulte os tópicos [Solução de problemas AWS CloudHSM](#) para obter ajuda. Para obter detalhes sobre os comandos da key_mgmt_util, consulte [Referência para comandos do AWS CloudHSM Key Management Utility](#).

Configurar AWS CloudHSM key_mgmt_util

Conclua a configuração a seguir antes de usar AWS CloudHSM key_mgmt_util (KMU).

Tópicos

- [Etapa 1. Inicie o AWS CloudHSM cliente](#)
- [Etapa 2. Iniciar key_mgmt_util](#)

Etapa 1. Inicie o AWS CloudHSM cliente

Antes de usar key_mgmt_util, você deve iniciar o cliente. AWS CloudHSM O cliente é um daemon que estabelece comunicação end-to-end criptografada com o HSMs em seu cluster. A ferramenta key_mgmt_util usa a conexão do cliente para se comunicar com o em seu cluster. HSMs Sem ela, a key_mgmt_util não funciona.

Para iniciar o AWS CloudHSM cliente

Use o comando a seguir para iniciar o AWS CloudHSM cliente.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```

RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe C:  
\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Etapa 2. Iniciar key_mgmt_util

Depois de iniciar o AWS CloudHSM cliente, use o comando a seguir para iniciar key_mgmt_util.

Amazon Linux

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Amazon Linux 2

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

CentOS 7

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

CentOS 8

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

RHEL 7

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

RHEL 8

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Ubuntu 16.04 LTS

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Ubuntu 18.04 LTS

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\key_mgmt_util.exe"
```

O prompt muda para Command: quando key_mgmt_util está em execução.

Se o comando falhar, como retornando uma mensagem `Daemon socket connection error`, tente [atualizar seu arquivo de configuração](#).

Faça login no HSMs em um AWS CloudHSM cluster usando KMU

Use o `loginHSM` comando em `key_mgmt_util` (KMU) para fazer login nos módulos de segurança de hardware (HSM) em um cluster. AWS CloudHSM O comando a seguir efetua login como um [usuário de criptografia \(CU\)](#) chamado `example_user`. A saída indica um login bem-sucedido para todos os três HSMs no cluster.

```
Command: loginHSM -u CU -s example_user -p <PASSWORD>
```

```
Cfm3LoginHSM returned: 0x00 : HSM Return: SUCCESS
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Veja a seguir a sintaxe do comando `loginHSM`.

```
Command: loginHSM -u <USER TYPE> -s <USERNAME> -p <PASSWORD>
```

Saia do HSMs em um AWS CloudHSM cluster usando KMU

Use o `logoutHSM` comando em `key_mgmt_util` (KMU) para sair dos módulos de segurança de hardware (HSM) em um cluster. AWS CloudHSM

```
Command: logoutHSM
```

```
Cfm3LogoutHSM returned: 0x00 : HSM Return: SUCCESS
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Pare o AWS CloudHSM `key_mgmt_util`

Use o `exit` comando para parar o AWS CloudHSM `key_mgmt_util`.

```
Command: exit
```

Instale e configure o AWS CloudHSM cliente para KMU (Linux)

Para interagir com o módulo de segurança de hardware (HSM) em seu AWS CloudHSM cluster usando o key_mgmt_util (KMU), você precisa do software cliente para Linux. AWS CloudHSM Você deve instalá-lo na instância do cliente do Linux EC2 criada anteriormente. Você também pode instalar um cliente se estiver usando o Windows. Para obter mais informações, consulte [Instale e configure o AWS CloudHSM cliente para KMU \(Windows\)](#).

Tarefas

- [Etapa 1. Instale as ferramentas AWS CloudHSM do cliente e da linha de comando](#)
- [Etapa 2. Editar a configuração do cliente](#)

Etapa 1. Instale as ferramentas AWS CloudHSM do cliente e da linha de comando

Conecte-se à sua instância cliente e execute os comandos a seguir para baixar e instalar as ferramentas do AWS CloudHSM cliente e da linha de comando.

Amazon Linux

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

RHEL 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

RHEL 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client_latest_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client_latest_u18.04_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_u18.04_amd64.deb
```

Etapa 2. Editar a configuração do cliente

Antes de usar o AWS CloudHSM cliente para se conectar ao seu cluster, você deve editar a configuração do cliente.

Para editar a configuração do cliente

1. Copie o certificado de emissão, [aquele usado para assinar o certificado do cluster](#), para o seguinte local na instância do cliente: `/opt/cloudhsm/etc/customerCA.crt`. É necessário ter permissões de usuário raiz da instância na instância do cliente para copiar o certificado para este local.
2. Use o comando [configure](#) a seguir para atualizar os arquivos de configuração das ferramentas do AWS CloudHSM cliente e da linha de comando, especificando o endereço IP do HSM em seu cluster. Para obter o endereço IP do HSM, visualize seu cluster no [AWS CloudHSM console](#) ou execute o [describe-clusters](#) AWS CLI comando. Na saída do comando, o endereço IP do HSM é o valor que está no campo `EniIp`. Se você tiver mais de um HSM, escolha o endereço IP de qualquer um deles HSMs; não importa qual.

```
sudo /opt/cloudhsm/bin/configure -a <IP address>
```

```
Updating server config in /opt/cloudhsm/etc/cloudhsm_client.cfg
```

```
Updating server config in /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

3. Acesse [Ative o cluster em AWS CloudHSM](#).

Instale e configure o AWS CloudHSM cliente para KMU (Windows)

Para trabalhar com um módulo de segurança de hardware (HSM) em seu AWS CloudHSM cluster no Windows usando o `key_mgmt_util` (KMU), você precisa do software cliente para Windows. AWS CloudHSM Você deve instalá-lo na instância do Windows Server criada anteriormente.

Para instalar (ou atualizar) o cliente e as ferramentas da linha de comando mais recentes do Windows

1. Conecte-se à sua instância do Windows Server.
2. Baixe a versão mais recente (AWSCloudHSMClient-latest.msi) na página de [downloads](#).
3. Vá até o local de download e execute o instalador (AWSCloudHSMClient-latest.msi) com privilégios administrativos.
4. Siga as instruções do instalador e escolha Fechar após a conclusão do instalador.
5. Copie o certificado de emissão autoassinado, [aquele usado para assinar o certificado do cluster](#) para a pasta C:\ProgramData\Amazon\CloudHSM.
6. Execute o seguinte comando para atualizar os arquivos de configuração. Certifique-se de parar e iniciar o cliente durante a reconfiguração, se você estiver atualizando-o:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM IP address>
```

7. Acesse [Ative o cluster em AWS CloudHSM](#).

Observações:

- Se você estiver atualizando o cliente, os arquivos de configuração existentes de instalações anteriores não serão substituídos.
- O instalador do AWS CloudHSM cliente para Windows registra automaticamente a API de criptografia: próxima geração (CNG) e o provedor de armazenamento de chaves (KSP). Para desinstalar o cliente, execute o instalador novamente e siga as instruções de desinstalação.
- Você também pode instalar um cliente se estiver usando o Linux. Para obter mais informações, consulte [Instale e configure o AWS CloudHSM cliente para KMU \(Linux\)](#).

Referência para comandos do AWS CloudHSM Key Management Utility

A ferramenta de linha de comando key_mgmt_util ajuda você a gerenciar chaves nos módulos de segurança de hardware (HSM) em seu AWS CloudHSM cluster, incluindo criar, excluir e localizar chaves e seus atributos. Ela inclui vários comandos, cada um descrito em detalhes neste tópico.

Para começar rapidamente, consulte [Começando com AWS CloudHSM key_mgmt_util](#). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#). Para obter informações sobre a ferramenta de linha de comando

cloudhsm_mgmt_util, que inclui comandos para gerenciar o HSM e os usuários em seu cluster, consulte [AWS CloudHSM Utilitário de gerenciamento \(CMU\)](#).

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Para listar todos os comandos key_mgmt_util, digite:

```
Command: help
```

Para obter ajuda para um comando específico da cloudhsm_mgmt_util, digite:

```
Command: <command-name> -h
```

Para encerrar sua sessão key_mgmt_util, digite:

```
Command: exit
```

Os seguintes tópicos descrevem comandos na key_mgmt_util.

 Note

Alguns comandos no key_mgmt_util e cloudhsm_mgmt_util têm os mesmos nomes. No entanto, os comandos normalmente têm sintaxe diferente, saída diferente e funcionalidade ligeiramente diferente.

Command	Description
aesWrapUnwrap	Criptografa e descriptografa o conteúdo de uma chave em um arquivo.
deleteKey	Exclui uma chave do HSMs.
Error2String	Obtém o erro que corresponde a um código de erro hexadecimal da cloudhsm_mgmt_util.
exit	Encerra a sessão da key_mgmt_util.

Command	Description
exportPrivateKey	Exporta uma cópia de uma chave privada de um HSM para um arquivo no disco.
exportPubKey	Exporta uma cópia de uma chave pública de um HSM para um arquivo.
exSymKey	Exporta uma cópia em texto simples de uma chave simétrica dos HSMs para um arquivo.
extractMaskedObject	Extrai uma chave de um HSM como um arquivo de objeto mascarado.
findKey	Procure chaves por valor de atributo de chave.
findSingleKey	Verifica se existe uma chave em todos os HSMs no cluster.
2 gênios DSAKey Pair	Gera um par de chaves do Algoritmo de Assinatura Digital (DSA) em seus HSMs.
2 gênios ECCKey Pair	Gera um par de chaves de criptografia de curva elíptica (ECC) em seus HSMs.
2 gênios RSAKey Pair	Gera um par de chaves assimétricas RSA em seus HSMs.
genSymKey	Gera uma chave simétrica em seus HSMs.
getAttribute	Obtém os valores dos atributos para uma chave do AWS CloudHSM e os grava em um arquivo.
getCaviumPrivChave	Cria uma versão em formato PEM falsa de uma chave privada e a exporta para um arquivo.
getCert	Recupera certificados de partições do HSM e os salva em um arquivo.

Command	Description
<u>getKeyInfo</u>	Obtém o usuário do HSM IDs dos usuários que podem usar a chave. Se a chave for controlada por quorum, ele obtém o número de usuários no quorum.
<u>help</u>	Exibe informações de ajuda sobre os comandos disponíveis na cloudhsm_mgmt_util.
<u>importPrivateKey</u>	Importa uma chave privada para um HSM.
<u>importPubKey</u>	Importa uma chave pública para um HSM.
<u>imSymKey</u>	Importa uma cópia em texto simples de uma chave simétrica de um arquivo para o HSM.
<u>insertMaskedObject</u>	Insere um objeto mascarado de um arquivo no disco em um HSM contido pelo cluster relacionado ao cluster de origem do objeto. Clusters relacionados são quaisquer clusters <u>gerados a partir de um backup do cluster de origem</u> .
<u>???</u>	Determina se um determinado arquivo contém ou não uma chave privada real ou uma chave PEM de exemplo.
<u>listAttributes</u>	Lista os atributos de uma AWS CloudHSM chave e as constantes que os representam.
<u>listUsers</u>	Obtém os usuários no HSMs, seu tipo e ID de usuário e outros atributos.
<u>loginHSM e logoutHSM</u>	Faça login e saia do HSMs em um cluster.
<u>setAttribute</u>	Converte uma chave de sessão em uma chave persistente.

Command	Description
sign	Gere uma assinatura para um arquivo usando uma chave privada escolhida.
unWrapKey	Importa uma chave encapsulada (criptografada) de um arquivo para HSMs o.
verify	Verifica se uma determinada chave foi usada para assinar um determinado arquivo.
wrapKey	Exporta uma cópia criptografada de uma chave do HSM para um arquivo.

Criptografe e descriptografe qualquer arquivo usando o KMU AWS CloudHSM

Use o `aesWrapUnwrap` comando em AWS CloudHSM `key_mgmt_util` para criptografar ou descriptografar o conteúdo de um arquivo no disco. Este comando foi projetado para encapsular e desencapsular chaves de criptografia, mas você pode usá-lo em qualquer arquivo que contenha menos de 4 KB (4096 bytes) de dados.

O `aesWrapUnwrap` usa o Agrupamento de chaves AES com preenchimento PKCS #5. Ele usa uma chave AES no HSM como a chave de encapsulamento ou desencapsulamento. Em seguida, ele grava o resultado em outro arquivo no disco.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```

aesWrapUnwrap -h

aesWrapUnwrap -m <wrap-unwrap mode>
               -f <file-to-wrap-unwrap>
               -w <wrapping-key-handle>
               [-i <wrapping-IV>]
               [-out <output-file>]

```

Exemplos

Esses exemplos mostram como usar `aesWrapUnwrap` para criptografar e descriptografar uma chave de criptografia em um arquivo.

Example: encapsular uma chave de criptografia

Esse comando usa `aesWrapUnwrap` para encapsular uma chave simétrica Triple DES que foi [exportada do HSM em texto simples](#) para o arquivo `3DES.key`. Você pode usar um comando semelhante para remover qualquer chave salva em um arquivo.

O comando usa o parâmetro `-m` com um valor de 1 para indicar o modo de enrolamento. Ele usa o parâmetro `-w` para especificar uma chave AES no HSM (identificador de chave 6) como a chave de encapsulamento. Ele grava a chave encapsulada resultante no arquivo `3DES.key.wrapped`.

A saída mostra que o comando foi bem-sucedido e que a operação usou o IV padrão, o que é preferido.

```
Command: aesWrapUnwrap -f 3DES.key -w 6 -m 1 -out 3DES.key.wrapped
```

```
Warning: IV (-i) is missing.
```

```
0xA6A6A6A6A6A6A6A6 is considered as default IV
```

```
result data:
```

```
49 49 E2 D0 11 C1 97 22
```

```
17 43 BD E3 4E F4 12 75
```

```
8D C1 34 CF 26 10 3A 8D
```

```
6D 0A 7B D5 D3 E8 4D C2
```

```
79 09 08 61 94 68 51 B7
```

```
result written to file 3DES.key.wrapped
```

```
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

Example: desencapsular uma chave de criptografia

Este exemplo mostra como usar `aesWrapUnwrap` para desempacotar (desencapsular) uma chave encapsulada (criptografada) em um arquivo. Você pode querer fazer uma operação como essa antes de importar uma chave para o HSM. Por exemplo, se você tentar usar o [imSymKey](#) comando para importar uma chave criptografada, ele retornará um erro porque a chave criptografada não tem o formato necessário para uma chave de texto simples desse tipo.

O comando desencapsula a chave no arquivo `3DES.key.wrapped` e grava o texto simples no arquivo `3DES.key.unwrapped`. O comando usa o parâmetro `-m` com um valor de `0` para indicar o modo de desencapsulamento. Ele usa o parâmetro `-w` para especificar uma chave AES no HSM (identificador de chave 6) como a chave de encapsulamento. Ele grava a chave encapsulada resultante no arquivo `3DES.key.unwrapped`.

```
Command: aesWrapUnwrap -m 0 -f 3DES.key.wrapped -w 6 -out 3DES.key.unwrapped
```

```
Warning: IV (-i) is missing.
```

```
0xA6A6A6A6A6A6A6A6 is considered as default IV
```

```
result data:
```

```
14 90 D7 AD D6 E4 F5 FA
```

```
A1 95 6F 24 89 79 F3 EE
```

```
37 21 E6 54 1F 3B 8D 62
```

```
result written to file 3DES.key.unwrapped
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-m

Especifica o modo. Para encapsular (criptografar) o conteúdo do arquivo, digite `1`; para desencapsular (descriptografar) o conteúdo do arquivo, digite `0`.

Obrigatório: Sim

-f

Especifica o arquivo a ser encapsulado. Insira um arquivo que contenha menos de 4 KB (4096 bytes) de dados. Essa operação foi projetada para encapsular e desencapsular chaves de criptografia.

Obrigatório: Sim

-w

Especifica a chave de encapsulamento. Insira o identificador de uma chave AES no HSM. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para criar uma chave de empacotamento, use [genSymKey](#) para gerar uma chave AES (tipo 31).

Obrigatório: Sim

-i

Especifica um valor inicial alternativo (IV) para o algoritmo. Use o valor padrão, a menos que você tenha uma condição especial que exija uma alternativa.

Padrão: 0xA6A6A6A6A6A6A6A6. O valor padrão é definido na especificação do algoritmo [AES Key Wrap](#).

Obrigatório: não

-out

Especifica um nome alternativo para o arquivo de saída que contém a chave encapsulada ou desencapsulada. O padrão é `wrapped_key` (para operações de encapsulamento) e `unwrapped_key` (para operações de desencapsulamento) no diretório local.

Se o arquivo existir, o `aesWrapUnwrap` o substituirá sem aviso prévio. Se o comando falhar, `aesWrapUnwrap` criará um arquivo de saída sem conteúdo.

Padrão: Para encapsulamento: `wrapped_key`. Para desencapsulamento: `unwrapped_key`.

Obrigatório: não

Tópicos relacionados

- [exSymKey](#)
- [imSymKey](#)
- [unWrapKey](#)
- [wrapKey](#)

Excluir uma AWS CloudHSM chave usando KMU

Use o `deleteKey` comando no AWS CloudHSM `key_mgmt_util` para excluir uma chave do módulo de segurança de hardware (HSM) em um cluster. AWS CloudHSM Você só pode excluir uma chave por vez. A exclusão de uma chave em um par de chaves não tem efeito na outra chave do par.

Somente o proprietário da chave pode excluí-la. Os usuários que compartilham a chave podem usá-la em operações criptográficas, mas não podem excluí-la.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
deleteKey -h
```

```
deleteKey -k
```

Exemplos

Esses exemplos mostram como usar `deleteKey` para excluir chaves do seu HSMs.

Example: excluir uma chave

Esse comando exclui a chave com identificador de chave 6. Quando o comando é bem-sucedido, `deleteKey` retorna mensagens de sucesso de cada HSM do cluster.

Command: **deleteKey -k 6**

```
Cfm3DeleteKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: excluir uma chave (falha)

Quando o comando falha porque nenhuma chave possui o identificador de chave especificado, `deleteKey` retorna uma mensagem de erro de identificador de objeto inválido.

Command: **deleteKey -k 252126**

```
Cfm3FindKey returned: 0xa8 : HSM Error: Invalid object handle is passed to this operation
```

Cluster Error Status

```
Node id 1 and err state 0x000000a8 : HSM Error: Invalid object handle is passed to this operation
```

```
Node id 2 and err state 0x000000a8 : HSM Error: Invalid object handle is passed to this operation
```

Quando o comando falhar porque o usuário atual não é o proprietário da chave, o comando retorna um erro de acesso negado.

```
Command: deleteKey -k 262152
```

```
Cfm3DeleteKey returned: 0xc6 : HSM Error: Key Access is denied.
```

Parâmetros

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-k

Especifica o identificador de chave da chave a ser excluída. Para encontrar os identificadores das chaves no HSM, use [findKey](#).

Obrigatório: Sim

Tópicos relacionados

- [findKey](#)

Descreva um AWS CloudHSM erro ao usar o KMU

Use o comando `Error2String` auxiliar no AWS CloudHSM `key_mgmt_util` para retornar o erro que corresponde a um código de erro hexadecimal `key_mgmt_util`. Você pode usar esse comando ao solucionar problemas com comandos e scripts.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar `key\_mgmt\_util`](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
Error2String -h
```

```
Error2String -r <response-code>
```

Exemplos

Estes exemplos mostram como usar `Error2String` para obter a string de erro para um código de erro da `key_mgmt_util`.

Example: Obter uma descrição do erro

Esse comando obtém a descrição do erro para o código de erro `0xdb`. A descrição explica que uma tentativa de fazer login na `key_mgmt_util` falhou porque o usuário possui o tipo de usuário errado. Somente usuários de criptografia (CU) podem fazer login na `key_mgmt_util`.

```
Command: Error2String -r 0xdb
```

```
Error Code db maps to HSM Error: Invalid User Type.
```

Example: Encontrar o código de erro

Este exemplo mostra onde encontrar o código de erro em um erro da `key_mgmt_util`. O código de erro, `0xc6`, aparece após a string: `Cfm3<command-name> returned: .`

Neste exemplo, [getKeyInfo](#) indica que o usuário atual (usuário 4) pode usar a chave em operações criptográficas. No entanto, quando o usuário tenta usar [deleteKey](#) para excluir a chave, o comando retorna o código de erro `0xc6`.

```
Command: deleteKey -k 262162
```

```
Cfm3DeleteKey returned: <0xc6> : HSM Error: Key Access is denied
```

```
Cluster Error Status
```

```
Command: getKeyInfo -k 262162
```



```
Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS
```

```
Owned by user 3
```

```
also, shared to following 1 user(s):
```

```
4
```

Se o erro 0xc6 for reportado, você poderá usar um comando `Error2String` como esse para procurar o erro. Nesse caso, o comando `deleteKey` falhou com um erro de acesso negado porque a chave é compartilhada com o usuário atual, mas possuída por um usuário diferente. Somente os proprietários de chaves têm permissão para excluir uma chave.

```
Command: Error2String -r 0xa8
```

```
Error Code c6 maps to HSM Error: Key Access is denied
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-r

Especifica um código de erro hexadecimal. É necessário o indicador de hexadecimal 0x.

Obrigatório: Sim

Saia da AWS CloudHSM KMU

Use o `exit` comando no `key_mgmt_util` para sair do AWS CloudHSM `key_mgmt_util`. Após a saída bem-sucedida, você retornará à sua linha de comando padrão.

Antes de executar qualquer comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#).

Sintaxe

```
exit
```

Parâmetros

Não há parâmetros para esse comando.

Tópicos relacionados

- [Iniciar key_mgmt_util](#)

Exportar uma AWS CloudHSM chave privada usando KMU

Use o `exportPrivateKey` comando no AWS CloudHSM `key_mgmt_util` para exportar uma chave privada assimétrica de um módulo de segurança de hardware (HSM) para um arquivo. O HSM não permite a exportação direta de chaves em texto não criptografado. O comando agrupa a chave privada usando uma chave de agrupamento AES que você especifica, descriptografa os bytes agrupados e copia a chave privada de texto não criptografado em um arquivo.

O comando `exportPrivateKey` não remove a chave do HSM nem altera os [atributos de chave](#) ou impede que você use a chave em outras operações criptográficas. É possível exportar a mesma chave várias vezes.

Você só pode exportar chaves privadas que têm o atributo `OBJ_ATTR_EXTRACTABLE` com o valor 1. Você deve especificar uma chave de agrupamento AES que tenha atributos de valor 1 `OBJ_ATTR_WRAP` e `OBJ_ATTR_DECRYPT`. Para encontrar os atributos de uma chave, use o comando [getAttribute](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
exportPrivateKey -h

exportPrivateKey -k <private-key-handle>
                  -w <wrapping-key-handle>
                  -out <key-file>
                  [-m <wrapping-mechanism>]
                  [-wk <wrapping-key-file>]
```

Exemplos

Este exemplo mostra como usar `exportPrivateKey` para exportar uma chave privada de um HSM.

Example: Export a Private Key

Esse comando exporta uma chave privada com o identificador 15 usando uma chave de encapsulamento com o identificador 16 para um arquivo PEM chamado `exportKey.pem`. Quando o comando é bem-sucedido, `exportPrivateKey` retorna uma mensagem de êxito.

```
Command: exportPrivateKey -k 15 -w 16 -out exportKey.pem
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
PEM formatted private key is written to exportKey.pem
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-k

Especifica o identificador de chave da chave privada a ser exportada.

Obrigatório: Sim

-w

Especifica o identificador de chave da chave de encapsulamento. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para determinar se uma chave pode ser usada como uma chave de encapsulamento, use [getAttribute](#) para obter o valor do atributo `OBJ_ATTR_WRAP` (262). Para criar uma chave de encapsulamento, use [genSymKey](#) a fim de criar uma chave AES (digite 31).

Se você usar o parâmetro `-wk` para especificar uma chave de desencapsulamento externa, a chave de encapsulamento `-w` será usada para encapsular, mas não para desencapsular, a chave durante a exportação.

Obrigatório: Sim

-out

Especifica o nome do arquivo no qual a chave privada exportada será gravada.

Obrigatório: Sim

-m

Especifica o mecanismo de encapsulamento com o qual encapsular a chave privada que está sendo exportada. O único valor válido é 4, que representa `NIST_AES_WRAP mechanism`.

Padrão: 4 (`NIST_AES_WRAP`)

Obrigatório: não

-wk

Especifica a chave a ser usada para desencapsular a chave que está sendo exportada. Insira o caminho e o nome de um arquivo que contém uma chave AES de texto simples.

Durante a inclusão desse parâmetro, `exportPrivateKey` usa a chave no arquivo `-w` para encapsular a chave que está sendo exportada e usa a chave especificada no parâmetro `-wk` para desencapsulá-la.

Padrão: use a chave de encapsulamento especificada no parâmetro `-w` para encapsular e desencapsular.

Obrigatório: não

Tópicos relacionados

- [importPrivateKey](#)
- [wrapKey](#)
- [unWrapKey](#)
- [genSymKey](#)

Exportar uma AWS CloudHSM chave pública usando KMU

Use o `exportPubKey` comando no AWS CloudHSM `key_mgmt_util` para exportar uma chave pública em um HSM para um arquivo. Você pode usá-lo para exportar chaves públicas que você gera em um HSM. Você também pode usar esse comando para exportar chaves públicas que foram importadas para um HSM, como as importadas com o comando [importPubKey](#).

A operação `exportPubKey` copia o material de chave em um arquivo especificado por você. Mas ele não remove a chave do HSM nem altera os [atributos de chave](#) ou impede que você use a chave em outras operações criptográficas. É possível exportar a mesma chave várias vezes.

Você só pode exportar chaves públicas que têm um valor `OBJ_ATTR_EXTRACTABLE` de 1. Para encontrar os atributos de uma chave, use o comando [getAttribute](#).

Para executar um comando da `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
exportPubKey -h

exportPubKey -k <public-key-handle>
               -out <key-file>
```

Exemplos

Este exemplo mostra como usar `exportPubKey` para exportar uma chave pública de um HSM.

Example: Export a Public Key

Esse comando exporta uma chave pública com identificador 10 para um arquivo chamado `public.pem`. Quando o comando é bem-sucedido, `exportPubKey` retorna uma mensagem de êxito.

```
Command: exportPubKey -k 10 -out public.pem

PEM formatted public key is written to public.pem

Cfm3ExportPubKey returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-k

Especifica o identificador de chave da chave pública a ser exportada.

Obrigatório: Sim

-out

Especifica o nome do arquivo no qual a chave pública exportada será gravada.

Obrigatório: Sim

Tópicos relacionados

- [importPubKey](#)
- [Gerar chaves](#)

Exportar uma cópia em texto simples de uma AWS CloudHSM chave usando KMU

Use o `exSymKey` comando na ferramenta AWS CloudHSM `key_mgmt_util` para exportar uma cópia em texto simples de uma chave simétrica do módulo de segurança de hardware (HSM) e salvá-la em um arquivo no disco. Para exportar uma cópia criptografada (encapsulada) de uma chave, use [wrapKey](#). Para importar uma chave de texto simples, como as que `exSymKey` exportam, use [imSymKey](#).

Durante o processo de exportação, `exSymKey` usa uma chave AES que você especifica (a chave de encapsulamento) para encapsular (criptografar) e depois desencapsular (descriptografar) a chave a ser exportada. No entanto, o resultado da operação de exportação é uma chave em texto simples (desencapsulada) no disco.

Somente o proprietário de uma chave, ou seja, o usuário CU que a criou, pode exportá-la. Os usuários que compartilham a chave podem usá-la em operações de criptografia, mas não podem exportá-la.

A operação `exSymKey` copia o material da chave para um arquivo que você especifica, mas não remove a chave do HSM, altera seus [atributos de chave](#) ou impede que você a use em operações criptográficas. É possível exportar a mesma chave várias vezes.

`exSymKey` exporta apenas chaves simétricas. Para exportar chaves públicas, use [exportPubKey](#). Para exportar chaves privadas, use [exportPrivateKey](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
exSymKey -h

exSymKey -k <key-to-export>
          -w <wrapping-key>
          -out <key-file>
          [-m 4]
          [-wk <unwrapping-key-file> ]
```

Exemplos

Esses exemplos mostram como usar `exSymKey` para exportar chaves simétricas que você possui do seu HSMs.

Example: Exportar uma chave simétrica 3DES

Esse comando exporta uma chave simétrica Triple DES (3DES) (identificador de chave 7). Ele usa uma chave AES existente (identificador de chave 6) no HSM como chave de encapsulamento. Em seguida, ele grava o texto simples da chave 3DES no arquivo `3DES.key`.

A saída mostra que a chave 7 (a tecla 3DES) foi encapsulada e desencapsulada com êxito e, em seguida, gravada no arquivo `3DES.key`.

⚠ Warning

Embora a saída diga que uma "Chave simétrica encapsulada" foi gravada no arquivo de saída, este último contém uma chave em texto simples (desencapsulada).

```
Command: exSymKey -k 7 -w 6 -out 3DES.key
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
Wrapped Symmetric Key written to file "3DES.key"
```

Example: Exportar com a chave de encapsulamento somente para sessão

Este exemplo mostra como usar uma chave que existe apenas na sessão como a chave de encapsulamento. Como a chave a ser exportada é encapsulada, imediatamente desencapsulada e fornecida como texto simples, não há necessidade de reter a chave de encapsulamento.

Esta série de comandos exporta uma chave AES com o identificador de chave 8 do HSM. Ela usa uma chave de sessão AES criada especialmente para esse propósito.

O primeiro comando é usado [genSymKey](#) para criar uma chave AES de 256 bits. Ele usa o parâmetro `-sess` para criar uma chave que existe apenas na sessão atual.

A saída mostra que o HSM cria a chave 262168.

```
Command: genSymKey -t 31 -s 32 -l AES-wrapping-key -sess
```

```
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Created. Key Handle: 262168
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Em seguida, o exemplo verifica se a chave 8, a chave a ser exportada, é uma chave simétrica extraível. Ele também verifica se a chave de encapsulamento, 262168, é uma chave AES que existe

apenas na sessão. Você pode usar o comando [findKey](#), mas esse exemplo exporta os atributos de ambas as chaves para arquivos e, em seguida, usa `grep` para encontrar valores de atributos relevantes no arquivo.

Esses comandos usam `getAttribute` com um valor `-a` de 512 (tudo) para obter todos os atributos para as chaves 8 e 262168. Para obter informações sobre os atributos de chaves, consulte [the section called “Referência de atributos de chave”](#).

```
getAttribute -o 8 -a 512 -out attributes/attr_8
getAttribute -o 262168 -a 512 -out attributes/attr_262168
```

Esses comandos usam `grep` para verificar os atributos da chave a ser exportada (chave 8) e da chave de encapsulamento somente da sessão (chave 262168).

```
// Verify that the key to be exported is a symmetric key.
$ grep -A 1 "OBJ_ATTR_CLASS" attributes/attr_8
OBJ_ATTR_CLASS
0x04

// Verify that the key to be exported is extractable.
$ grep -A 1 "OBJ_ATTR_KEY_TYPE" attributes/attr_8
OBJ_ATTR_EXTRACTABLE
0x00000001

// Verify that the wrapping key is an AES key
$ grep -A 1 "OBJ_ATTR_KEY_TYPE" attributes/attr_262168
OBJ_ATTR_KEY_TYPE
0x1f

// Verify that the wrapping key is a session key
$ grep -A 1 "OBJ_ATTR_TOKEN" attributes/attr_262168
OBJ_ATTR_TOKEN
0x00

// Verify that the wrapping key can be used for wrapping
$ grep -A 1 "OBJ_ATTR_WRAP" attributes/attr_262168
OBJ_ATTR_WRAP
0x00000001
```

Por último, usamos um comando `exSymKey` para exportar a chave 8 usando a chave de sessão (chave 262168) como a chave de encapsulamento.

Quando a sessão terminar, a chave 262168 não existirá mais.

```
Command: exSymKey -k 8 -w 262168 -out aes256_H8.key
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
Wrapped Symmetric Key written to file "aes256_H8.key"
```

Example: Usar uma chave de desencapsulamento externa

Este exemplo mostra como usar uma chave de desencapsulamento externa para exportar uma chave do HSM.

Ao exportar uma chave do HSM, você especifica uma chave AES no HSM para ser a chave de encapsulamento. Por padrão, essa chave de encapsulamento é usada para encapsular e desencapsular a chave a ser exportada. No entanto, você pode usar o parâmetro `-wk` para instruir `exSymKey` a usar uma chave externa em um arquivo no disco para desencapsulamento. Ao fazer isso, a chave especificada pelo parâmetro `-w` encapsula a chave de destino e a chave no arquivo especificado pelo parâmetro `-wk` desencapsula a chave.

Como a chave de encapsulamento deve ser uma chave AES, que é simétrica, a chave de encapsulamento no HSM e a chave no disco devem ter o mesmo material de chave. Para fazer isso, você deve importar a chave de encapsulamento para o HSM ou exportá-la do HSM antes da operação de exportação.

Esse exemplo cria uma chave fora do HSM e a importa para o HSM. Ele usa a cópia interna da chave para encapsular uma chave simétrica que está sendo exportada e usa a cópia dessa chave no arquivo para desencapsulá-la.

O primeiro comando usa o OpenSSL para gerar uma chave AES de 256 bits. Ele salva a chave no arquivo `aes256-forImport.key`. O comando OpenSSL não retorna saída, mas você pode usar vários comandos para confirmar seu sucesso. Este exemplo usa a ferramenta `wc` (contagem de palavras), que confirma que o arquivo tem 32 bytes de dados.

```
$ openssl rand -out keys/aes256-forImport.key 32
```

```
$ wc keys/aes256-forImport.key
0  2 32 keys/aes256-forImport.key
```

Esse comando usa o comando [imSymKey](#) para importar a chave AES do arquivo `aes256-forImport.key` para o HSM. Quando o comando é concluído, a chave existe no HSM com o identificador de chave 262167 e no arquivo `aes256-forImport.key`.

```
Command: imSymKey -f keys/aes256-forImport.key -t 31 -l aes256-imported -w 6
```

```
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Unwrapped. Key Handle: 262167
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Esse comando usa a chave em uma operação de exportação. O comando usa `exSymKey` para exportar a chave 21, uma chave AES de 192 bits. Para encapsular a chave, ele usa a chave 262167, que é a cópia que foi importada para o HSM. Para desempacotar a chave, ela usa o mesmo material de chave no arquivo `aes256-forImport.key`. Quando o comando for concluído, a chave 21 será exportada para o arquivo `aes192_h21.key`.

```
Command: exSymKey -k 21 -w 262167 -out aes192_H21.key -wk aes256-forImport.key
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Wrapped Symmetric Key written to file "aes192_H21.key"
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-k

Especifica o identificador de chave da chave a ser exportada. Esse parâmetro é obrigatório. Insira o identificador de chave de uma chave simétrica que você possui. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para verificar se uma chave pode ser exportada, use o comando [getAttribute](#) para obter o valor do atributo OBJ_ATTR_EXTRACTABLE, que é representado pela constante 354. Além disso, você pode exportar apenas as chaves que você possui. Para encontrar o proprietário de uma chave, use o [getKeyInfo](#) comando.

Obrigatório: Sim

-w

Especifica o identificador de chave da chave de encapsulamento. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Uma chave de encapsulamento é uma chave no HSM que é usada para criptografar (encapsular) e depois descriptografar (desencapsular) a chave a ser exportada. Somente as chaves AES podem ser usadas como chaves de encapsulamento.

Você pode usar qualquer chave do AES (de qualquer tamanho) como uma chave de encapsulamento. Como a chave de encapsulamento encapsula e depois desencapsula imediatamente a chave de destino, você pode usar como chave AES somente de sessão como uma chave de encapsulamento. Para determinar se uma chave pode ser usada como uma chave de encapsulamento, use [getAttribute](#) para obter o valor do atributo OBJ_ATTR_WRAP, que é representado pela constante 262. Para criar uma chave de encapsulamento, use [genSymKey](#) a fim de criar uma chave AES (digite 31).

Se você usar o parâmetro -wk para especificar uma chave de desencapsulamento externa, a chave de encapsulamento -w será usada para encapsular, mas não para desencapsular, a chave durante a exportação.

 Note

A chave 4 representa uma chave interna sem suporte. Recomendamos usar uma chave AES criada e gerenciada por você como a chave de encapsulamento.

Obrigatório: Sim

-out

Especifica o caminho e o nome do arquivo de saída. Quando o comando é bem-sucedido, esse arquivo contém a chave exportada em texto simples. Se o arquivo já existir, o comando o sobrescreverá sem aviso prévio.

Obrigatório: Sim

-m

Especifica o mecanismo de encapsulamento. O único valor válido é 4, que representa o mecanismo NIST_AES_WRAP.

Obrigatório: não

Padrão: 4

-wk

Use a chave AES no arquivo especificado para desencapsular a chave que está sendo exportada. Insira o caminho e o nome de um arquivo que contém uma chave AES de texto simples.

Durante a inclusão desse parâmetro, `exSymKey` usa a chave no arquivo no HSM especificada pelo parâmetro `-w` para encapsular a chave que está sendo exportada e usa a chave no arquivo `-wk` para desencapsulá-la. Os valores dos parâmetros `-w` e `-wk` devem ser resolvidos para a mesma chave de texto simples.

Obrigatório: não

Padrão: use a chave de encapsulamento no HSM para desencapsular.

Tópicos relacionados

- [genSymKey](#)
- [imSymKey](#)
- [wrapKey](#)

Extraia uma AWS CloudHSM chave usando KMU

Use o `extractMaskedObject` comando no AWS CloudHSM `key_mgmt_util` para extrair uma chave de um módulo de segurança de hardware (HSM) e salvá-la em um arquivo como um objeto

mascarado. Objetos mascarados são objetos clonados que só podem ser usados após serem inseridos novamente no cluster original usando o comando [insertMaskedObject](#). Você só pode inserir um objeto mascarado no mesmo cluster do qual ele foi gerado ou em um clone desse cluster. Isso inclui quaisquer versões clonadas do cluster geradas ao [copiar um backup entre regiões](#) e [usar esse backup para criar um novo cluster](#).

Objetos mascarados são uma maneira eficiente de descarregar e sincronizar chaves, incluindo chaves nonextractable (isto é, chaves que têm um `OBJ_ATTR_EXTRACTABLE` valor de 0). [Dessa forma, as chaves podem ser sincronizadas com segurança entre clusters relacionados em diferentes regiões sem a necessidade de atualizar o AWS CloudHSM arquivo de configuração.](#)

Important

Na inserção, os objetos mascarados são descriptografados e recebem um identificador de chave diferente do identificador de chave da chave original. Um objeto mascarado inclui todos os metadados associados à chave original, inclusive atributos, informações de propriedade e compartilhamento e configurações de quórum. Se você precisa sincronizar as chaves em clusters em um aplicativo, use [syncKey](#) cloudhsm_mgmt_util instead.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM. O comando `extractMaskedObject` pode ser usado pelo CU que possui a chave ou qualquer CO.

Sintaxe

```
extractMaskedObject -h

extractMaskedObject -o <object-handle>
                    -out <object-file>
```

Exemplos

Este exemplo mostra como usar `extractMaskedObject` para extrair uma chave de um HSM como um objeto mascarado.

Example: Extract a Masked Object

Esse comando extrai um objeto mascarado de um HSM a partir de uma chave com identificador 524295 e a salva como um arquivo chamado `maskedObj`. Quando o comando é bem-sucedido, `extractMaskedObject` retorna uma mensagem de êxito.

```
Command: extractMaskedObject -o 524295 -out maskedObj
```

```
Object was masked and written to file "maskedObj"
```

```
Cfm3ExtractMaskedObject returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-o

Especifica o identificador da chave a ser extraída como um objeto mascarado.

Obrigatório: Sim

-out

Especifica o nome do arquivo no qual o objeto mascarado será salvo.

Obrigatório: Sim

Tópicos relacionados

- [insertMaskedObject](#)
- [syncKey](#)
- [Copiar um backup entre regiões](#)
- [Criando um AWS CloudHSM cluster a partir de um backup anterior](#)

Pesquise AWS CloudHSM chaves por atributos usando KMU

Use o findKey comando no AWS CloudHSM key_mgmt_util para pesquisar chaves pelos valores dos atributos da chave. Quando uma chave corresponder a todos os critérios que você definir, findKey retornará o identificador de chave. Sem parâmetros, findKey retorna os identificadores de chave de todas as chaves que você pode usar no HSM. Para encontrar os valores de atributo de uma determinada chave, use [getAttribute](#).

Como todos os comandos de key_mgmt_util, findKey é específico do usuário. Ele retorna apenas as chaves que o usuário atual pode usar em operações criptográficas. Isso inclui chaves que o usuário atual possui e as chaves que foram compartilhadas com ele.

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
findKey -h

findKey [-c <key class>]
        [-t <key type>]
        [-l <key label>]
        [-id <key ID>]
        [-sess (0 | 1)]
        [-u <user-ids>]
        [-m <modulus>]
        [-kcv <key_check_value>]
```

Exemplos

Esses exemplos mostram como usar findKey para encontrar e identificar chaves em seu HSMs.

Example: Encontrar todas as chaves

Esse comando encontra todas as chaves para o usuário atual no HSM. A saída inclui chaves que o usuário possui e compartilha, e todas as chaves públicas no HSMs.

Para obter os atributos de uma chave com um identificador de chave específico, use [getAttribute](#). Para determinar se o usuário atual possui ou compartilha uma chave específica, use [getKeyInfo](#) ou [findAllKeys](#) em cloudhsm_mgmt_util.

Command: **findKey**

Total number of keys present 13

number of keys matched from start index 0::12

6, 7, 524296, 9, 262154, 262155, 262156, 262157, 262158, 262159, 262160, 262161, 262162

Cluster Error Status

Node id 1 and err state 0x00000000 : HSM Return: SUCCESS

Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS

Example: Encontrar chaves por tipo, usuário e sessão

Este comando encontra chaves AES persistentes que o usuário atual e o usuário 3 podem usar. (O usuário 3 pode usar outras chaves que o usuário atual não pode ver.)

Command: **findKey -t 31 -sess 0 -u 3**

Example: Encontrar chaves por classe e rótulo

Este comando encontra todas as chaves públicas para o usuário atual com o rótulo 2018-sept.

Command: **findKey -c 2 -l 2018-sept**

Example: Encontrar chaves RSA por módulo

Este comando encontra as chaves RSA (tipo 0) para o usuário atual que foram criadas usando o módulo no arquivo m4.txt.

Command: **findKey -t 0 -m m4.txt**

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-t

Encontra chaves do tipo especificado. Insira a constante que representa a classe de chave. Por exemplo, para encontrar chaves 3DES, digite -t 21.

Valores válidos:

- 0: [RSA](#)
- 1: [DSA](#)
- 3: [EC](#)
- 16: [GENERIC_SECRET](#)
- 18: [RC4](#)
- 21: [Triple DES \(3DES\)](#)
- 31: [AES](#)

Obrigatório: não

-c

Localiza chaves na classe especificada. Insira a constante que representa a classe de chave. Por exemplo, para encontrar chaves públicas, digite -c 2.

Valores válidos para cada tipo de chave:

- 2: Pública. Essa classe contém as chaves públicas dos pares de chaves públicas-privadas.
- 3: Privada. Essa classe contém as chaves privadas dos pares de chaves públicas-privadas.
- 4: Secreta. Essa classe contém todas as chaves simétricas.

Obrigatório: não

-l

Encontra chaves com o rótulo especificado. Digite o rótulo exato. Você não pode usar caracteres curinga ou expressões regulares no valor -l.

Obrigatório: não

-id

Encontra a chave com o ID especificado. Digite a string de ID exata. Você não pode usar caracteres curinga ou expressões regulares no valor -id.

Obrigatório: não

-sess

Localiza chaves por status da sessão. Para encontrar as chaves válidas apenas na sessão atual, digite 1. Para localizar chaves persistentes, digite 0.

Obrigatório: não

-u

Encontra as chaves dos usuários especificados e o compartilhamento de usuários atual. Digite uma lista separada por vírgulas de usuários do HSM IDs, como ou. -u 3 -u 4,7 Para encontrar o número IDs de usuários em um HSM, use [ListUsers](#).

Quando você especifica um ID de usuário, findKey retorna as chaves para esse usuário.

Quando você especifica vários usuários IDs, findKey retorna as chaves que todos os usuários especificados podem usar.

Como findKey simplesmente retorna as chaves que o usuário atual pode usar, os resultados de -u são sempre idênticos às, ou um subconjunto das, chaves do usuário atual. Para obter todas as chaves pertencentes a ou compartilhadas com qualquer usuário, os agentes criptográficos (COs) podem usar [findAllKey](#) em cloudhsm_mgmt_util.

Obrigatório: não

-m

Localiza as chaves que foram criadas usando o módulo RSA no arquivo especificado. Digite o caminho para o arquivo que armazena o módulo.

-m especifica o arquivo binário que contém o módulo RSA a ser correspondido (opcional).

Obrigatório: não

-kcv

Encontra chaves com o valor de verificação de chave especificado.

O key check value (KCV – valor de verificação de chave) é um hash de 3 bytes ou soma de verificação de uma chave gerada quando o HSM importa ou gera uma chave. Você também pode calcular um KCV fora do HSM, como depois de exportar uma chave. Em seguida, é possível comparar os valores do KCV para confirmar a identidade e a integridade da chave. Para obter o KCV de uma chave, use [getAttribute](#).

AWS CloudHSM usa o seguinte método padrão para gerar um valor de verificação de chave:

- Chaves simétricas: primeiros 3 bytes do resultado da criptografia de um bloco zero com a chave.
- Pares de chaves assimétricas: primeiros 3 bytes do hash SHA-1 da chave pública.
- Chaves HMAC: o KCV para chaves HMAC não é suportado no momento.

Obrigatório: não

Output

A saída findKey mostra o número total de chaves correspondentes e seus identificadores de chave.

```
Command: findKey
Total number of keys present 10

number of keys matched from start index 0::9
6, 7, 8, 9, 10, 11, 262156, 262157, 262158, 262159

Cluster Error Status
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS

Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Tópicos relacionados

- [findSingleKey](#)
- [getKeyInfo](#)
- [getAttribute](#)
- [findAllKey](#) em cloudhsm_mgmt_util
- [Referência de atributos de chave](#)

Verifique uma AWS CloudHSM chave usando KMU

Use o findSingleKey comando na ferramenta AWS CloudHSM key_mgmt_util para verificar se existe uma chave em todos os módulos de segurança de hardware (HSM) no cluster. AWS CloudHSM

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
findSingleKey -h  
  
findSingleKey -k <key-handle>
```

Exemplo

Example

Esse comando verifica se a chave 252136 existe em todos os três HSMs no cluster.

```
Command: findSingleKey -k 252136  
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS  
  
Cluster Error Status  
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS  
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS  
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-k

Especifica o identificador da chave no HSM. Esse parâmetro é obrigatório.

Para encontrar os identificadores de chave, use o comando [findKey](#).

Obrigatório: Sim

Tópicos relacionados

- [findKey](#)
- [getKeyInfo](#)

- [getAttribute](#)

Gere um par de chaves AWS CloudHSM DSA usando KMU

Use o `genDSAKeyPair` comando na ferramenta AWS CloudHSM `key_mgmt_util` para gerar um par de chaves do Algoritmo de [Assinatura Digital \(DSA\)](#) em seus módulos de segurança de hardware (HSM). Você deve especificar o comprimento do módulo; o comando gera o valor do módulo. Você também pode atribuir um ID, compartilhar a chave com outros usuários do HSM e criar chaves não extraíveis que expiram quando a sessão termina. Quando o comando é bem-sucedido, ele retorna os identificadores de chave que o HSM atribui às chaves públicas e privadas. Você pode usar os identificadores de chave para identificar as chaves para outros comandos.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Tip

Para encontrar os atributos de uma chave criada por você, como tipo, comprimento, rótulo e ID, use [getAttribute](#). Para encontrar as chaves de um usuário específico, use [getKeyInfo](#). Para encontrar chaves com base em seus valores de atributos, use [FindKey](#).

Sintaxe

```
genDSAKeyPair -h

genDSAKeyPair -m <modulus length>
               -l <label>
               [-id <key ID>]
               [-min_srv <minimum number of servers>]
               [-m_value <0..8>]
               [-nex]
               [-sess]
               [-timeout <number of seconds> ]
               [-u <user-ids>]
               [-attest]
```

Exemplos

Esses exemplos mostram como usar `genDSAKeyPair` para criar um par de chaves DSA.

Example: Criar um par de chaves DSA

Esse comando cria um par de chaves DSA com um rótulo DSA. A saída mostra que o identificador da chave pública é 19 e o identificador da chave privada é 21.

```
Command: genDSAKeyPair -m 2048 -l DSA
```

```
Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3GenerateKeyPair:    public key handle: 19    private key handle: 21
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Criar um par de chaves DSA somente de sessão

Este comando cria um par de chaves DSA que é válido apenas na sessão atual. O comando atribui um ID exclusiva de `DSA_temp_pair`, além do rótulo (não exclusivo) requerido. Você pode querer criar um par de chaves como esse para assinar e verificar um token somente de sessão. A saída mostra que o identificador da chave pública é 12 e o identificador da chave privada é 14.

```
Command: genDSAKeyPair -m 2048 -l DSA-temp -id DSA_temp_pair -sess
```

```
Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3GenerateKeyPair:    public key handle: 12    private key handle: 14
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Para confirmar que o par de chaves existe apenas na sessão, use o parâmetro `-sess` de [findKey](#) com o valor de 1 (true).

```
Command: findKey -sess 1
```

```
Total number of keys present 2
```

```
number of keys matched from start index 0::1
```

```
12, 14
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: Criar um par de chaves DSA compartilhadas e não extraíveis

Esse comando cria um par de chaves DSA. A chave privada é compartilhada com outros três usuários e não pode ser exportada do HSM. Chaves públicas podem ser usadas por qualquer usuário e sempre podem ser extraídas.

```
Command: genDSAKeyPair -m 2048 -l DSA -id DSA_shared_pair -nex -u 3,5,6

Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS

Cfm3GenerateKeyPair:    public key handle: 11    private key handle: 19

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Criar um par de chaves com controle de quorum

Esse comando cria um par de chaves DSA com o rótulo DSA-mV2. O comando usa o parâmetro `-u` para compartilhar a chave privada com os usuários 4 e 6. Ele usa o parâmetro `-m_value` para exigir um quorum de pelo menos duas aprovações para quaisquer operações criptográficas que usam a chave privada. O comando também usa o parâmetro `-attest` para verificar a integridade do firmware no qual o par de chaves é gerado.

A saída mostra que o comando gera uma chave pública com o identificador de chave 12 e uma chave privada com o identificador de chave 17 e que a verificação de atestado no firmware do cluster foi aprovada.

```
Command: genDSAKeyPair -m 2048 -l DSA-mV2 -m_value 2 -u 4,6 -attest

Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS

Cfm3GenerateKeyPair:    public key handle: 12    private key handle: 17

Attestation Check : [PASS]

Cluster Error Status
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```



```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Esse comando é usado [getKeyInfo](#) na chave privada (identificador da chave 17). A saída confirma que a chave pertence ao usuário atual (usuário 3) e que é compartilhada com os usuários 4 e 6 (e nenhum outro). A saída também mostra que a autenticação de quorum está habilitada e que o tamanho do quorum é dois.

```
Command:  getKeyInfo -k 17

Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

Owned by user 3

also, shared to following 2 user(s):

        4
        6
2 Users need to approve to use/manage this key
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-m

Especifica o comprimento do módulo em bits. O único valor válido é 2048.

Obrigatório: Sim

-l

Especifica o rótulo definido pelo usuário para o par de chaves. Digite uma string. O mesmo rótulo se aplica às duas chaves do par. O tamanho máximo para `label` é de 127 caracteres.

É possível usar qualquer frase que ajude a identificar a chave. Como o rótulo não precisa ser exclusivo, é possível usá-lo para agrupar e categorizar chaves.

Obrigatório: Sim

-id

Especifica um rótulo definido pelo usuário para o par de chaves. Digite uma string exclusiva no cluster. O padrão é uma string vazia. O ID que você especifica se aplica às duas chaves do par.

Padrão: sem valor de ID.

Obrigatório: não

-min_srv

Especifica o número mínimo HSMs no qual a chave é sincronizada antes que o valor do - timeout parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de min_srv como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-m_value

Especifica o número de usuários que devem aprovar qualquer operação criptográfica que use a chave privada no par. Digite um valor de 0 até 8.

Esse parâmetro estabelece um requisito de autenticação de quórum para a chave privada. O valor padrão, 0, desativa o atributo de autenticação de quórum para a chave. Quando a autenticação de quórum está habilitada, o número especificado de usuários deve assinar um token para aprovar operações criptográficas que usam a chave privada e operações que compartilham ou descompartilham a chave privada.

Para encontrar m_value a chave, use [getKeyInfo](#).

Esse parâmetro é válido somente quando o parâmetro -u no comando compartilha a chave com usuários o suficiente para atender ao requisito de m_value.

Padrão: 0

Obrigatório: não

-nex

Torna a chave privada não extraível. A chave privada gerada não pode ser [exportada do HSM](#). Chaves públicas sempre podem ser extraídas.

Padrão: as chaves pública e privada no par de chaves podem ser extraídas.

Obrigatório: não

-sess

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [setAttribute](#).

Padrão: a chave é persistente.

Obrigatório: não

-timeout

Especifica por quanto tempo (em segundos) o comando espera que uma chave seja sincronizada com o número HSMs especificado pelo parâmetro. `min_srv`

Este parâmetro é válido somente quando o parâmetro `min_srv` também é usado no comando.

Padrão: sem limite de tempo. O comando espera indefinidamente e retorna somente quando a chave é sincronizada com o número mínimo de servidores.

Obrigatório: não

-u

Compartilha a chave privada no par com os usuários especificados. Esse parâmetro dá permissão a outros usuários de criptografia do HSM (CUs) para usar a chave privada em

operações criptográficas. As chaves públicas podem ser usadas por qualquer usuário sem compartilhamento.

Digite uma lista separada por vírgulas do usuário do HSM IDs, como -. u 5,6 Não inclua o ID do usuário atual do HSM. [Para encontrar o usuário IDs do HSM CUs no HSM, use ListUsers.](#) Em seguida, para compartilhar ou descompartilhar uma chave existente, use [shareKey](#) em `cloudhsm_mgmt_util`.

Padrão: somente o usuário atual pode utilizar a chave importada.

Obrigatório: não

-attest

Executa uma verificação de integridade que verifica se o firmware no qual o cluster é executado não foi adulterado.

Padrão: sem verificação de atestado.

Obrigatório: não

Tópicos relacionados

- [2 gênios RSAKey Pair](#)
- [genSymKey](#)
- [2 gênios ECCKey Pair](#)

Gere um par de chaves AWS CloudHSM ECC usando KMU

Use o `genECCKeyPair` comando na ferramenta AWS CloudHSM `key_mgmt_util` para gerar um par de chaves de [criptografia de curva elíptica \(ECC\) em seus módulos de segurança de hardware](#) (HSM). Ao executar o comando `genECCKeyPair`, você deve especificar o identificador de curva elíptica e um rótulo para o par de chaves. Você também pode compartilhar a chave privada com outros usuários CU, criar chaves não extraíveis, chaves com controle de quorum e chaves que expiram quando a sessão termina. Quando o comando é bem-sucedido, ele retorna os identificadores de chave que o HSM atribui às chaves ECC públicas e privadas. Você pode usar os identificadores de chave para identificar as chaves para outros comandos.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Tip

Para encontrar os atributos de uma chave criada por você, como tipo, comprimento, rótulo e ID, use [getAttribute](#). Para encontrar as chaves de um usuário específico, use [getKeyInfo](#). Para encontrar chaves com base em seus valores de atributos, use [FindKey](#).

Sintaxe

```
genECCKeyPair -h

genECCKeyPair -i <EC curve id>
                -l <label>
                [-id <key ID>]
                [-min_srv <minimum number of servers>]
                [-m_value <0..8>]
                [-nex]
                [-sess]
                [-timeout <number of seconds> ]
                [-u <user-ids>]
                [-attest]
```

Exemplos

Os exemplos a seguir mostram como usar `genECCKeyPair` para criar pares de chaves ECC em seu HSMs.

Example: Criar e examinar um par de chaves ECC

Esse comando usa uma curva elíptica `NID_secp384r1` e um rótulo `ecc14` para criar um par de chaves ECC. A saída mostra que o identificador da chave privada é 262177 e o identificador da chave pública é 262179. O rótulo aplica-se a chaves públicas e privadas.

Command: **genECCKeyPair -i 14 -l ecc14**

```
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3GenerateKeyPair:    public key handle: 262179    private key handle: 262177
```

```
Cluster Error Status
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Depois de gerar a chave, você pode examinar seus atributos. Use [getAttribute](#) para gravar todos os atributos (representados pela constante 512) da nova chave privada ECC no arquivo `attr_262177`.

```
Command: getAttribute -o 262177 -a 512 -out attr_262177  
got all attributes of size 529 attr cnt 19  
Attributes dumped into attr_262177
```

```
Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS
```

Em seguida, use o comando `cat` para visualizar o conteúdo do arquivo de atributo `attr_262177`. A saída mostra que a chave é uma chave privada de curva elíptica que pode ser usada para assinar, mas não para criptografar, descriptografar, encapsular, desencapsular ou verificar. A chave é persistente e exportável.

```
$ cat attr_262177  
  
OBJ_ATTR_CLASS  
0x03  
OBJ_ATTR_KEY_TYPE  
0x03  
OBJ_ATTR_TOKEN  
0x01  
OBJ_ATTR_PRIVATE  
0x01  
OBJ_ATTR_ENCRYPT  
0x00  
OBJ_ATTR_DECRYPT  
0x00  
OBJ_ATTR_WRAP  
0x00  
OBJ_ATTR_UNWRAP  
0x00  
OBJ_ATTR_SIGN  
0x01  
OBJ_ATTR_VERIFY  
0x00  
OBJ_ATTR_LOCAL  
0x01  
OBJ_ATTR_SENSITIVE  
0x01
```

```

OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
ecc2
OBJ_ATTR_ID

OBJ_ATTR_VALUE_LEN
0x0000008a
OBJ_ATTR_KCV
0xbbb32a
OBJ_ATTR_MODULUS
044a0f9d01d10f7437d9fa20995f0cc742552e5ba16d3d7e9a65a33e20ad3e569e68eb62477a9960a87911e6121d112
OBJ_ATTR_MODULUS_BITS
0x0000019f

```

Example Usar uma curva EEC inválida

Esse comando tenta criar um par de chaves ECC usando uma curva NID_X9_62_prime192v1. Como essa curva elíptica não é válida para o modo FIPS HSMs, o comando falha. A mensagem informa que um servidor no cluster não está disponível, mas isso normalmente não indica um problema com o HSMs no cluster.

Command: **genECCKeyPair -i 1 -l ecc1**

```

Cfm3GenerateKeyPair returned: 0xb3 : HSM Error: This operation violates the
current configured/FIPS policies

```

```

Cluster Error Status

```

```

Node id 0 and err state 0x30000085 : HSM CLUSTER ERROR: Server in cluster is
unavailable

```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-i

Especifica o identificador da curva elíptica. Insira um identificador.

Valores válidos:

- 2: NID_X9_62_prime256v1
- 14: NID_secp384r1
- 16: NID_secp256k1

Obrigatório: Sim

-l

Especifica o rótulo definido pelo usuário para o par de chaves. Digite uma string. O mesmo rótulo se aplica às duas chaves do par. O tamanho máximo para `label` é de 127 caracteres.

É possível usar qualquer frase que ajude a identificar a chave. Como o rótulo não precisa ser exclusivo, é possível usá-lo para agrupar e categorizar chaves.

Obrigatório: Sim

-id

Especifica um rótulo definido pelo usuário para o par de chaves. Digite uma string exclusiva no cluster. O padrão é uma string vazia. O ID que você especifica se aplica às duas chaves do par.

Padrão: sem valor de ID.

Obrigatório: não

-min_srv

Especifica o número mínimo HSMs no qual a chave é sincronizada antes que o valor do `-timeout` parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de `min_srv` como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-m_value

Especifica o número de usuários que devem aprovar qualquer operação criptográfica que use a chave privada no par. Digite um valor de 0 até 8.

Esse parâmetro estabelece um requisito de autenticação de quórum para a chave privada. O valor padrão, 0, desativa o atributo de autenticação de quórum para a chave. Quando a autenticação de quórum está habilitada, o número especificado de usuários deve assinar um token para aprovar operações criptográficas que usam a chave privada e operações que compartilham ou descompartilham a chave privada.

Para encontrar `m_value` a chave, use [getKeyInfo](#).

Esse parâmetro é válido somente quando o parâmetro `-u` no comando compartilha a chave com usuários o suficiente para atender ao requisito de `m_value`.

Padrão: 0

Obrigatório: não

`-nex`

Torna a chave privada não extraível. A chave privada gerada não pode ser [exportada do HSM](#). Chaves públicas sempre podem ser extraídas.

Padrão: as chaves pública e privada no par de chaves podem ser extraídas.

Obrigatório: não

`-sess`

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [setAttribute](#).

Padrão: a chave é persistente.

Obrigatório: não

`-timeout`

Especifica por quanto tempo (em segundos) o comando espera que uma chave seja sincronizada com o número HSMs especificado pelo parâmetro. `min_srv`

Este parâmetro é válido somente quando o parâmetro `min_srv` também é usado no comando.

Padrão: sem limite de tempo. O comando espera indefinidamente e retorna somente quando a chave é sincronizada com o número mínimo de servidores.

Obrigatório: não

-u

Compartilha a chave privada no par com os usuários especificados. Esse parâmetro dá permissão a outros usuários de criptografia do HSM (CUs) para usar a chave privada em operações criptográficas. As chaves públicas podem ser usadas por qualquer usuário sem compartilhamento.

Digite uma lista separada por vírgulas do usuário do HSM IDs, como `-. u 5,6`. Não inclua o ID do usuário atual do HSM. [Para encontrar o usuário IDs do HSM CUs no HSM, use `ListUsers`.](#) Em seguida, para compartilhar ou descompartilhar uma chave existente, use [shareKey](#) em `cloudhsm_mgmt_util`.

Padrão: somente o usuário atual pode utilizar a chave importada.

Obrigatório: não

-attest

Executa uma verificação de integridade que verifica se o firmware no qual o cluster é executado não foi adulterado.

Padrão: sem verificação de atestado.

Obrigatório: não

Tópicos relacionados

- [genSymKey](#)
- [2 gênios RSAKey Pair](#)
- [2 gênios DSAKey Pair](#)

Gere um par de chaves AWS CloudHSM RSA usando KMU

[Use o `genRSAKeyPair` comando na ferramenta AWS CloudHSM `key\_mgmt\_util` para gerar um par de chaves assimétrico RSA.](#) Você especifica o tipo de chave, o comprimento do módulo e um

expoente público. O comando gera um módulo do comprimento especificado e cria o par de chaves. Você pode atribuir um ID, compartilhar a chave com outros usuários do HSM, criar chaves e chaves não extraíveis que expiram quando a sessão termina. Quando o comando for bem-sucedido, ele retornará um identificador de chave que o HSM atribui à chave. Você pode usar o identificador de chave para identificar as chaves para outros comandos.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Tip

Para encontrar os atributos de uma chave criada por você, como tipo, comprimento, rótulo e ID, use [getAttribute](#). Para encontrar as chaves de um usuário específico, use [getKeyInfo](#). Para encontrar chaves com base em seus valores de atributos, use [FindKey](#).

Sintaxe

```
genRSAKeyPair -h

genRSAKeyPair -m <modulus length>
               -e <public exponent>
               -l <label>
               [-id <key ID>]
               [-min_srv <minimum number of servers>]
               [-m_value <0..8>]
               [-nex]
               [-sess]
               [-timeout <number of seconds> ]
               [-u <user-ids>]
               [-attest]
```

Exemplos

Esses exemplos mostram como usar `genRSAKeyPair` para criar pares de chaves assimétricas em seu HSMs

Example: Criar e examinar um par de chaves RSA

Esse comando cria um par de chaves RSA com um módulo de 2048 bits e um expoente de 65537. A saída mostra que o identificador de chave pública é 2100177 e o identificador de chave particular é 2100426.

```
Command: genRSAKeyPair -m 2048 -e 65537 -l rsa_test
```

```
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS
```

```
      Cfm3GenerateKeyPair:      public key handle: 2100177      private key handle:
2100426
```

```
Cluster Status:
```

```
Node id 0 status: 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 status: 0x00000000 : HSM Return: SUCCESS
```

O próximo comando usa [getAttribute](#) para obter os atributos da chave pública que acabamos de criar. Ele grava a saída no arquivo `attr_2100177`. Ele é seguido por um comando `cat` que obtém o conteúdo do arquivo de atributo. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Os valores hexadecimais resultantes confirmam que se trata de uma chave pública (`OBJ_ATTR_CLASS 0x02`) com um tipo de RSA (`OBJ_ATTR_KEY_TYPE 0x00`). Você pode usar essa chave pública para criptografar (`OBJ_ATTR_ENCRYPT 0x01`), mas não para descriptografar (`OBJ_ATTR_DECRYPT 0x00`). Os resultados também incluem o comprimento da chave (512, `0x200`), o módulo, o comprimento do módulo (2048, `0x800`) e o expoente público (65537, `0x10001`).

```
Command: getAttribute -o 2100177 -a 512 -out attr_2100177
```

```
Attribute size: 801, count: 26
```

```
Written to: attr_2100177 file
```

```
Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS
```

```
$ cat attr_2100177
```

```
OBJ_ATTR_CLASS
```

```
0x02
```

```
OBJ_ATTR_KEY_TYPE
```

```
0x00
```

```
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
0x01
OBJ_ATTR_ENCRYPT
0x01
OBJ_ATTR_DECRYPT
0x00
OBJ_ATTR_WRAP
0x01
OBJ_ATTR_UNWRAP
0x00
OBJ_ATTR_SIGN
0x00
OBJ_ATTR_VERIFY
0x01
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x00
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
rsa_test
OBJ_ATTR_ID

OBJ_ATTR_VALUE_LEN
0x00000200
OBJ_ATTR_KCV
0xc51c18
OBJ_ATTR_MODULUS
0xbb9301cc362c1d9724eb93da8adab0364296bde7124a241087d9436b9be57e4f7780040df03c2c
1c0fe6e3b61aa83c205280119452868f66541bbbffacbbe787b8284fc81deaeef2b8ec0ba25a077d
6983c77a1de7b17cbe8e15b203868704c6452c2810344a7f2736012424cf0703cf15a37183a1d2d0
97240829f8f90b063dd3a41171402b162578d581980976653935431da0c1260bfe756d85dca63857
d9f27a541676cb9c7def0ef6a2a89c9b9304bcac16fdf8183c0a555421f9ad5dfef534cf26b65873
970cdf1a07484f1c128b53e10209cc6f7ac308669112968c81a5de408e7f644fe58b1a9ae1286fec
b3e4203294a96fae06f8f0db7982cb5d7f
OBJ_ATTR_MODULUS_BITS
0x00000800
OBJ_ATTR_PUBLIC_EXPONENT
0x010001
OBJ_ATTR_TRUSTED
0x00
```

```

OBJ_ATTR_WRAP_WITH_TRUSTED
0x00
OBJ_ATTR_DESTROYABLE
0x01
OBJ_ATTR_DERIVE
0x00
OBJ_ATTR_ALWAYS_SENSITIVE
0x00
OBJ_ATTR_NEVER_EXTRACTABLE
0x00

```

Example: Gerar um par de chaves RSA compartilhado

Este comando gera um par de chaves RSA e compartilha a chave privada com o usuário 4, outro CU no HSM. O comando usa o parâmetro `m_value` para exigir pelo menos duas aprovações antes que a chave privada no par possa ser usada em uma operação criptográfica. Ao usar o parâmetro `m_value`, você também deve usar `-u` no comando, e `m_value` não pode exceder o número total de usuários (número de valores no `-u` + proprietário).

Command: **genRSAKeyPair -m 2048 -e 65537 -l rsa_mofn -id rsa_mv2 -u 4 -m_value 2**

```
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3GenerateKeyPair:    public key handle: 27    private key handle: 28
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-m

Especifica o comprimento do módulo em bits. O valor mínimo é 2048.

Obrigatório: Sim

-p

Especifica o expoente público. O valor deve ser um número ímpar maior que ou igual a 65537.

Obrigatório: Sim

-l

Especifica o rótulo definido pelo usuário para o par de chaves. Digite uma string. O mesmo rótulo se aplica às duas chaves do par. O tamanho máximo para `label` é de 127 caracteres.

É possível usar qualquer frase que ajude a identificar a chave. Como o rótulo não precisa ser exclusivo, é possível usá-lo para agrupar e categorizar chaves.

Obrigatório: Sim

-id

Especifica um rótulo definido pelo usuário para o par de chaves. Digite uma string exclusiva no cluster. O padrão é uma string vazia. O ID que você especifica se aplica às duas chaves do par.

Padrão: sem valor de ID.

Obrigatório: não

-min_srv

Especifica o número mínimo HSMs no qual a chave é sincronizada antes que o valor do `-timeout` parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de `min_srv` como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-m_value

Especifica o número de usuários que devem aprovar qualquer operação criptográfica que use a chave privada no par. Digite um valor de 0 até 8.

Esse parâmetro estabelece um requisito de autenticação de quórum para a chave privada. O valor padrão, 0, desativa o atributo de autenticação de quórum para a chave. Quando a autenticação de quórum está habilitada, o número especificado de usuários deve assinar um token para aprovar operações criptográficas que usam a chave privada e operações que compartilham ou descompartilham a chave privada.

Para encontrar `m_value` a chave, use [getKeyInfo](#).

Esse parâmetro é válido somente quando o parâmetro `-u` no comando compartilha a chave com usuários o suficiente para atender ao requisito de `m_value`.

Padrão: 0

Obrigatório: não

`-nex`

Torna a chave privada não extraível. A chave privada gerada não pode ser [exportada do HSM](#). Chaves públicas sempre podem ser extraídas.

Padrão: as chaves pública e privada no par de chaves podem ser extraídas.

Obrigatório: não

`-sess`

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [setAttribute](#).

Padrão: a chave é persistente.

Obrigatório: não

`-timeout`

Especifica por quanto tempo (em segundos) o comando espera que uma chave seja sincronizada com o número HSMs especificado pelo parâmetro. `min_srv`

Este parâmetro é válido somente quando o parâmetro `min_srv` também é usado no comando.

Padrão: sem limite de tempo. O comando espera indefinidamente e retorna somente quando a chave é sincronizada com o número mínimo de servidores.

Obrigatório: não

-u

Compartilha a chave privada no par com os usuários especificados. Esse parâmetro dá permissão a outros usuários de criptografia do HSM (CUs) para usar a chave privada em operações criptográficas. As chaves públicas podem ser usadas por qualquer usuário sem compartilhamento.

Digite uma lista separada por vírgulas do usuário do HSM IDs, como `-. u 5,6`. Não inclua o ID do usuário atual do HSM. [Para encontrar o usuário IDs do HSM CUs no HSM, use `ListUsers`.](#) Em seguida, para compartilhar ou descompartilhar uma chave existente, use [shareKey](#) em `cloudhsm_mgmt_util`.

Padrão: somente o usuário atual pode utilizar a chave importada.

Obrigatório: não

-attest

Executa uma verificação de integridade que verifica se o firmware no qual o cluster é executado não foi adulterado.

Padrão: sem verificação de atestado.

Obrigatório: não

Tópicos relacionados

- [genSymKey](#)
- [2 gênios DSAKey Pair](#)
- [2 gênios ECCKey Pair](#)

Gere uma chave AWS CloudHSM simétrica usando KMU

Use o `genSymKey` comando na ferramenta AWS CloudHSM `key_mgmt_util` para gerar uma chave simétrica em seus módulos de segurança de hardware (HSM). Você pode especificar o tipo de chave

e o tamanho, atribuir um ID e um rótulo e compartilhar a chave com outros usuários do HSM. Você também pode criar chaves não extraíveis e chaves que expiram quando a sessão termina. Quando o comando for bem-sucedido, ele retornará um identificador de chave que o HSM atribui à chave. Você pode usar o identificador de chave para identificar as chaves para outros comandos.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar `key\_mgmt\_util`](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
genSymKey -h

genSymKey -t <key-type>
          -s <key-size>
          -l <label>
          [-id <key-ID>]
          [-min_srv <minimum-number-of-servers>]
          [-m_value <0..8>]
          [-nex]
          [-sess]
          [-timeout <number-of-seconds> ]
          [-u <user-ids>]
          [-attest]
```

Exemplos

Esses exemplos mostram como usar `genSymKey` para criar chaves simétricas em seu HSMs.

Tip

Para usar as chaves criadas com esses exemplos para operações HMAC, você deve definir `OBJ_ATTR_SIGN` e `OBJ_ATTR_VERIFY` para `TRUE` depois de gerar a chave. Para definir esses valores, use `setAttribute` no CloudHSM Management Utility (CMU). Para obter mais informações, consulte [setAttribute](#).

Example: Gerar uma chave de AES

Esse comando cria uma chave AES de 256 bits com um rótulo `aes256`. A saída mostra que o identificador da nova chave é 6.

```
Command: genSymKey -t 31 -s 32 -l aes256
```

```
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Created. Key Handle: 6
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Criar uma chave de sessão

Este comando cria uma chave AES de 192 bits não extraível que é válida apenas na sessão atual. Você talvez queira criar uma chave como essa para encapsular (e depois imediatamente desencapsular) uma chave que está sendo exportada.

```
Command: genSymKey -t 31 -s 24 -l tmpAES -id wrap01 -nex -sess
```

Example: Retornar rapidamente

Esse comando cria uma chave de 512 bytes genérica com um rótulo de `IT_test_key`. O comando não espera que a chave seja sincronizada com todos HSMs no cluster. Em vez disso, ele é retornado assim que a chave é criada em qualquer HSM (`-min_srv 1`) ou em 1 segundo (`-timeout 1`), o que for mais curto. Se a chave não estiver sincronizada com o número mínimo especificado HSMs antes da expiração do tempo limite, ela não será gerada. Você pode querer usar um comando como esse em um script que cria várias chaves, como o loop `for` no exemplo a seguir.

```
Command: genSymKey -t 16 -s 512 -l IT_test_key -min_srv 1 -timeout 1
```

```
$ for i in {1..30};
  do /opt/cloudhsm/bin/key_mgmt_util singlecmd loginHSM -u CU -s example_user -p
example_pwd genSymKey -l aes -t 31 -s 32 -min_srv 1 -timeout 1;
done;
```

Example: Criar uma chave genérica autorizada de quorum

Esse comando cria uma chave secreta genérica de 2048 bits com o rótulo `generic-mv2`. O comando usa o parâmetro `-u` para compartilhar a chave com outro CU, o usuário 6. Ele usa o parâmetro `-m_value` para exigir um quorum de pelo menos duas aprovações para quaisquer operações criptográficas que usam a chave. O comando também usa o parâmetro `-attest` para verificar a integridade do firmware no qual a chave é gerada.

A saída mostra que o comando gerou uma chave com o identificador de chave 9 e que a verificação de comprovação no firmware de cluster foi aprovada.

```
Command: genSymKey -t 16 -s 2048 -l generic-mV2 -m_value 2 -u 6 -  
attest  
  
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS  
  
Symmetric Key Created. Key Handle: 9  
  
Attestation Check : [PASS]  
  
Cluster Error Status  
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS  
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Criar e examinar uma chave

Esse comando cria uma chave DES Tripla com um rótulo 3DES_shared e um ID de IT-02. A chave pode ser usada pelo usuário atual e pelos usuários 4 e 5. O comando falhará se o ID não for exclusivo no cluster ou se o usuário atual for o usuário 4 ou 5.

A saída mostra que a nova chave tem o identificador de chave 7.

```
Command: genSymKey -t 21 -s 24 -l 3DES_shared -id IT-02 -u 4,5  
  
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS  
  
Symmetric Key Created. Key Handle: 7  
  
Cluster Error Status  
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Para verificar se a nova chave 3DES é de propriedade do usuário atual e compartilhada com os usuários 4 e 5, use [getKeyInfo](#). O comando usa o identificador que foi atribuído à nova chave (Key Handle: 7).

A saída confirma que a chave é de propriedade do usuário 3 e está compartilhada com os usuários 4 e 5.

```
Command: getKeyInfo -k 7
```

```
Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS
```

```
Owned by user 3
```

```
also, shared to following 2 user(s):
```

```
4, 5
```

Para confirmar as outras propriedades da chave, use [getAttribute](#). O primeiro comando usa `getAttribute` para obter todos os atributos (`-a 512`) do identificador de chave 7 (`-o 7`). Ele os grava no arquivo `attr_7`. O segundo comando usa `cat` para obter o conteúdo do arquivo `attr_7`.

Esse comando confirma que a chave 7 é uma chave simétrica de 192 bits (`OBJ_ATTR_VALUE_LEN 0x00000018` ou 24 bytes) 3DES (`OBJ_ATTR_KEY_TYPE 0x15`) (`OBJ_ATTR_CLASS 0x04`) com rótulo 3DES_shared (`OBJ_ATTR_LABEL 3DES_shared`) e ID IT_02 (`OBJ_ATTR_ID IT-02`). A chave é persistente (`OBJ_ATTR_TOKEN 0x01`) e extraível (`OBJ_ATTR_EXTRACTABLE 0x01`) e pode ser usada para criptografia, descryptografia e encapsulamento.

Tip

Para encontrar os atributos de uma chave criada por você, como tipo, comprimento, rótulo e ID, use [getAttribute](#). Para encontrar as chaves de um usuário específico, use [getKeyInfo](#). Para encontrar chaves com base em seus valores de atributos, use [FindKey](#).

Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

```
Command: getAttribute -o 7 -a 512 -out attr_7
```

```
got all attributes of size 444 attr cnt 17  
Attributes dumped into attr_7 file
```

```
Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS
```

```
$ cat attr_7
```

```
OBJ_ATTR_CLASS
0x04
OBJ_ATTR_KEY_TYPE
0x15
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
0x01
OBJ_ATTR_ENCRYPT
0x01
OBJ_ATTR_DECRYPT
0x01
OBJ_ATTR_WRAP
0x00
OBJ_ATTR_UNWRAP
0x00
OBJ_ATTR_SIGN
0x00
OBJ_ATTR_VERIFY
0x00
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x01
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
3DES_shared
OBJ_ATTR_ID
IT-02
OBJ_ATTR_VALUE_LEN
0x00000018
OBJ_ATTR_KCV
0x59a46e
```

 Tip

Para usar as chaves criadas com esses exemplos para operações HMAC, você deve definir OBJ_ATTR_SIGN e OBJ_ATTR_VERIFY para TRUE depois de gerar a chave. Para definir esses valores, use `setAttribute` no CMU. Para obter mais informações, consulte [setAttribute](#).

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-t

Especifica o tipo da chave simétrica. Insira a constante que representa o tipo de chave. Por exemplo, para criar uma chave AES, digite -t 31.

Valores válidos:

- 16: [GENERIC_SECRET](#). Uma chave secreta genérica é uma matriz de bytes que não está em conformidade com nenhum padrão específico, como os requisitos para uma chave AES.
- 18: [RC4](#). RC4 as chaves não são válidas no modo FIPS HSMs
- 21: [Triple DES \(3DES\)](#). De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).
- 31: [AES](#)

Obrigatório: Sim

-s

Especifica o tamanho da chave em bytes. Por exemplo, para criar uma chave de 192 bits, digite 24.

Valores válidos para cada tipo de chave:

- AES: 16 (128 bits), 24 (192 bits), 32 (256 bits)
- 3DES: 24 (192 bits)
- Segredo genérico: <3584 (28.672 bits)

Obrigatório: Sim

-l

Especifica o rótulo da chave privada definida pelo usuário para a chave. Digite uma string.

É possível usar qualquer frase que ajude a identificar a chave. Como o rótulo não precisa ser exclusivo, é possível usá-lo para agrupar e categorizar chaves.

Obrigatório: Sim

-attest

Executa uma verificação de integridade que verifica se o firmware no qual o cluster é executado não foi adulterado.

Padrão: sem verificação de atestado.

Obrigatório: não

-id

Especifica o identificador da chave definida pelo usuário. Digite uma string exclusiva no cluster. O padrão é uma string vazia.

Padrão: sem valor de ID.

Obrigatório: não

-min_srv

Especifica o número mínimo HSMs no qual a chave é sincronizada antes que o valor do - timeout parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de `min_srv` como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-m_value

Especifica o número de usuários que devem aprovar qualquer operação criptográfica que use a chave importada. Digite um valor de 0 até 8.

Esse parâmetro estabelece um requisito de autenticação de quórum para a chave. O valor padrão, 0, desativa o atributo de autenticação de quórum para a chave. Quando a autenticação

de quórum está habilitada, o número especificado de usuários deve assinar um token para aprovar operações criptográficas que usam a chave e operações que compartilham ou descompartilham a chave.

Para encontrar `m_value` a chave, use [getKeyInfo](#).

Esse parâmetro é válido somente quando o parâmetro `-u` no comando compartilha a chave com usuários o suficiente para atender ao requisito de `m_value`.

Padrão: 0

Obrigatório: não

`-nex`

Torna a chave não extraível. A chave gerada não pode ser [exportada do HSM](#).

Padrão: a chave é extraível.

Obrigatório: não

`-sess`

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [setAttribute](#).

Padrão: a chave é persistente.

Obrigatório: não

`-timeout`

Especifica por quanto tempo (em segundos) o comando espera que uma chave seja sincronizada com o número HSMs especificado pelo parâmetro. `min_srv`

Este parâmetro é válido somente quando o parâmetro `min_srv` também é usado no comando.

Padrão: sem limite de tempo. O comando espera indefinidamente e retorna somente quando a chave é sincronizada com o número mínimo de servidores.

Obrigatório: não

-u

Compartilha a chave com o usuário especificado. Esse parâmetro dá permissão a outros usuários de criptografia do HSM (CUs) para usar essa chave em operações criptográficas.

Digite uma lista separada por vírgulas do usuário do HSM IDs, como -. u 5,6 Não inclua o ID do usuário atual do HSM. [Para encontrar o usuário IDs do HSM CUs no HSM, use ListUsers.](#) Em seguida, para compartilhar ou descompartilhar uma chave existente, use [shareKey](#) em `cloudhsm_mgmt_util`.

Padrão: somente o usuário atual pode utilizar a chave.

Obrigatório: não

Tópicos relacionados

- [exSymKey](#)
- [2 gênios RSAKey Pair](#)
- [2 gênios DSAKey Pair](#)
- [2 gênios ECCKey Pair](#)
- [setAttribute](#)

Obtenha um AWS CloudHSM atributo-chave usando KMU

Use o `getAttribute` comando no AWS CloudHSM `key_mgmt_util` para gravar um ou todos os valores de atributo de uma chave em um arquivo. AWS CloudHSM Se o atributo que você especificar não existir para o tipo de chave, como o módulo de uma chave AES, `getAttribute` retornará um erro.

Atributos de chave são propriedades de uma chave. Eles incluem características, como o tipo de chave, a classe, o rótulo e o ID, e valores que representam ações que você pode realizar com a chave, como criptografar, descriptografar, encapsular, assinar e verificar.

Você só pode usar `getAttribute` nas chaves que você possui e em chaves que são compartilhadas com você. Você pode executar esse comando ou o comando [getAttribute](#) em `cloudhsm_mgmt_util`,

que obtém um valor de atributo de uma chave de tudo HSMs em um cluster e o grava em stdout ou em um arquivo.

Para obter uma lista de atributos e das constantes que os representam, use o comando [listAttributes](#). Para alterar os valores de atributo de chaves existentes, use [setAttribute](#) na key_mgmt_util e [setAttribute](#) na cloudhsm_mgmt_util. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
getAttribute -h

getAttribute -o <key handle>
               -a <attribute constant>
               -out <file>
```

Exemplos

Esses exemplos mostram como usar getAttribute para obter os atributos das chaves em seu HSMs.

Example: Obter o tipo de chave

Esse exemplo obtém o tipo de chave, como uma chave AES, 3DES ou genérica, ou um par de chaves RSA ou de curva elíptica.

O primeiro comando executa [listAttributes](#), que obtém os principais atributos e as constantes que os representam. A saída mostra que a constante para o tipo de chave é 256. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Command: **listAttributes**

Description

=====

The following are all of the possible attribute values for getAttributes.

OBJ_ATTR_CLASS	= 0
OBJ_ATTR_TOKEN	= 1
OBJ_ATTR_PRIVATE	= 2

OBJ_ATTR_LABEL	= 3
OBJ_ATTR_KEY_TYPE	= 256
OBJ_ATTR_ID	= 258
OBJ_ATTR_SENSITIVE	= 259
OBJ_ATTR_ENCRYPT	= 260
OBJ_ATTR_DECRYPT	= 261
OBJ_ATTR_WRAP	= 262
OBJ_ATTR_UNWRAP	= 263
OBJ_ATTR_SIGN	= 264
OBJ_ATTR_VERIFY	= 266
OBJ_ATTR_LOCAL	= 355
OBJ_ATTR_MODULUS	= 288
OBJ_ATTR_MODULUS_BITS	= 289
OBJ_ATTR_PUBLIC_EXPONENT	= 290
OBJ_ATTR_VALUE_LEN	= 353
OBJ_ATTR_EXTRACTABLE	= 354
OBJ_ATTR_KCV	= 371

O segundo comando executa `getAttribute`. Ele solicita o tipo de chave (atributo 256) para o identificador de chave 524296 e o grava no arquivo `attribute.txt`.

```
Command: getAttribute -o 524296 -a 256 -out attribute.txt
Attributes dumped into attribute.txt file
```

O comando final obtém o conteúdo do arquivo de chave. A saída revela que o tipo de chave é 0x15 ou 21, que é uma chave Triple DES (3DES). Para as definições dos valores da classe e tipo, consulte a [Referência de atributos de chave](#).

```
$ cat attribute.txt
OBJ_ATTR_KEY_TYPE
0x00000015
```

Example: Obter todos os atributos de uma chave

Este comando obtém todos os atributos da chave com o identificador de chave 6 e os grava no arquivo `attr_6`. Ele usa um valor de atributo de 512, que representa todos os atributos.

```
Command: getAttribute -o 6 -a 512 -out attr_6

got all attributes of size 444 attr cnt 17
Attributes dumped into attribute.txt file
```

```
Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS>
```

Esse comando mostra o conteúdo de um arquivo de atributo de amostra com todos os valores de atributos. Entre os valores, ele informa que a chave é uma chave AES de 256 bits com um ID de test_01 e um rótulo de aes256. A chave é extraível e persistente, ou seja, não é uma chave somente de sessão. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

```
$ cat attribute.txt
```

```
OBJ_ATTR_CLASS
0x04
OBJ_ATTR_KEY_TYPE
0x15
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
0x01
OBJ_ATTR_ENCRYPT
0x01
OBJ_ATTR_DECRYPT
0x01
OBJ_ATTR_WRAP
0x01
OBJ_ATTR_UNWRAP
0x01
OBJ_ATTR_SIGN
0x00
OBJ_ATTR_VERIFY
0x00
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x01
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
aes256
OBJ_ATTR_ID
test_01
OBJ_ATTR_VALUE_LEN
0x00000020
OBJ_ATTR_KCV
```

```
0x1a4b31
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-o

Especifica o identificador da chave de destino. Você pode especificar apenas uma chave em cada comando. Para obter o identificador de chave de uma chave, use [findKey](#).

Além disso, você deve ter a chave especificada ou ela deve ser compartilhada com você. Para encontrar os usuários de uma chave, use [getKeyInfo](#).

Obrigatório: Sim

-a

Identifica o atributo. Insira uma constante que represente um atributo, ou 512, que represente todos os atributos. Por exemplo, para obter o tipo de chave, digite 256, que é a constante para o atributo OBJ_ATTR_KEY_TYPE.

Para listar os atributos e suas constantes, use [listAttributes](#). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Obrigatório: Sim

-out

Grava a saída no arquivo especificado. Digite um caminho de arquivo. Você não pode gravar a saída em stdout.

Se o arquivo especificado existir, getAttribute substitui o arquivo sem aviso prévio.

Obrigatório: Sim

Tópicos relacionados

- [getAttribute](#) em cloudhsm_mgmt_util

- [listAttributes](#)
- [setAttribute](#)
- [findKey](#)
- [Referência de atributos de chave](#)

Exporte uma AWS CloudHSM chave para um formato PEM falso usando KMU

Use o comando `getCaviumPrivKey` no AWS CloudHSM `key_mgmt_util` para exportar uma chave privada de um módulo de segurança de hardware (HSM) em formato PEM falso. O arquivo PEM falso, que não contém o material real da chave privada, mas faz referência à chave privada no HSM, pode então ser usado para estabelecer o SSL/TLS descarregamento do seu servidor web para. AWS CloudHSM Para obter mais informações, consulte [SSL/TLS Offload on Linux using Tomcat](#) ou [SSL/TLS Offload on Linux using NGINX or Apache](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
getCaviumPrivKey -h

getCaviumPrivKey -k <private-key-handle>
                  -out <fake-PEM-file>
```

Exemplos

Este exemplo mostra como usar `getCaviumPrivKey` para exportar uma chave privada em um formato PEM falso.

Example: Export a Fake PEM File

Este comando cria e exporta uma versão PEM falsa de uma chave privada com identificador 15 e salva-a em um arquivo chamado `cavKey.pem`. Quando o comando é bem-sucedido, `exportPrivateKey` retorna uma mensagem de êxito.

```
Command: getCaviumPrivKey -k 15 -out cavKey.pem
```

```
Private Key Handle is written to cavKey.pem in fake PEM format
```

```
getCaviumPrivKey returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-k

Especifica o identificador de chave da chave privada a ser exportada no formato PEM falso.

Obrigatório: Sim

-out

Especifica o nome do arquivo no qual a chave PEM falsa será gravada.

Obrigatório: Sim

Tópicos relacionados

- [importPrivateKey](#)
- [Descarregamento de SSL/TLS no Linux usando Tomcat](#)
- [Descarregamento de SSL/TLS no Linux usando NGINX ou Apache](#)

Obtenha certificados de partição HSM usando AWS CloudHSM KMU

Use o `getCert` comando no AWS CloudHSM `key_mgmt_util` para recuperar os certificados de partição de um módulo de segurança de hardware (HSM) e salvá-los em um arquivo. Ao executar o comando, você designa o tipo de certificado a ser recuperado. Para fazer isso, você usa um dos números inteiros correspondentes, conforme descrito na seção [Parâmetros](#) a seguir. Para saber sobre a função de cada um desses certificados, consulte [Verificar a identidade do HSM](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
getCert -h

getCert -f <file-name>
        -t <certificate-type>
```

Exemplo

Este exemplo mostra como usar `getCert` para recuperar um certificado raiz do cliente do cluster e salvá-lo como um arquivo.

Example: recuperar um certificado raiz do cliente

Este comando exporta um certificado raiz do cliente (representado pelo inteiro 4) e salva-o em um arquivo chamado `userRoot.crt`. Quando o comando é bem-sucedido, `getCert` retorna uma mensagem de êxito.

```
Command: getCert -f userRoot.crt -s 4

Cfm3GetCert() returned 0 :HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-f

Especifica o nome do arquivo no qual o certificado recuperado será salvo.

Obrigatório: Sim

-s

Um inteiro que especifica o tipo do certificado de partição a ser recuperado. Os inteiros e seus tipos de certificado correspondentes são os seguintes:

- 1 – certificado raiz do fabricante
- 2 – certificado de hardware do fabricante
- 4 – certificado raiz do cliente
- 8 – certificado do cluster (assinado pelo certificado raiz do cliente)
- 16 – certificado do cluster (encadeado ao certificado raiz do fabricante)

Obrigatório: Sim

Tópicos relacionados

- [Verificar a identidade do HSM](#)

Obtenha os usuários de uma AWS CloudHSM chave usando o KMU

Use o `getKeyInfo` comando no AWS CloudHSM `key_mgmt_util` para retornar o usuário do módulo de segurança de hardware (HSM) IDs dos usuários que podem usar a chave, incluindo o proprietário e os usuários criptográficos (UC) com os quais a chave é compartilhada. Quando a autenticação de quorum está habilitada em uma chave, `getKeyInfo` também retorna o número de usuários que devem aprovar as operações criptográficas que usam essa chave. Você pode executar `getKeyInfo` somente nas chaves que possui e em chaves que são compartilhadas com você.

Quando você executa `getKeyInfo` em chaves públicas, `getKeyInfo` retorna apenas o proprietário da chave, mesmo que todos os usuários do HSM possam usar a chave pública. Para encontrar o usuário HSM IDs dos usuários em seu HSMs, use [ListUsers](#). Para encontrar as chaves de um usuário específico, use [findKey](#) -u.

Você possui as chaves que cria. Você pode compartilhar uma chave com outros usuários ao criá-la. Em seguida, para compartilhar ou descompartilhar uma chave existente, use [shareKey](#) em `cloudhsm_mgmt_util`.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
getKeyInfo -h
```

```
getKeyInfo -k <key-handle>
```

Exemplos

Esses exemplos mostram como usar `getKeyInfo` para obter informações sobre os usuários de uma chave.

Example: obter usuários para uma chave simétrica

Esse comando obtém os usuários que podem usar a chave AES (simétrica) com o identificador de chave 9. A saída mostra que o usuário 3 possui a chave e a compartilhou com o usuário 4.

Command: **getKeyInfo -k 9**

```
Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS
```

```
Owned by user 3
```

```
also, shared to following 1 user(s):
```

```
4
```

Example: obter os usuários para um par de chaves assimétricas

Esses comandos usam `getKeyInfo` para obter os usuários que podem usar as chaves em um par de chaves RSA (assimétrico). A chave pública possui o identificador de chave 21. A chave privada possui o identificador de chave 20.

Quando você `getKeyInfo` executa a chave privada (20), ela retorna o proprietário da chave (3) e os usuários criptográficos (CUs) 4 e 5, com os quais a chave é compartilhada.

Command: **getKeyInfo -k 20**

```
Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS
```

```
Owned by user 3
```

```
also, shared to following 2 user(s):
```

```
4
```

```
5
```

Quando você executa `getKeyInfo` na chave pública (21), ele retorna apenas o proprietário da chave (3).

```
Command: getKeyInfo -k 21
```

```
Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS
```

```
Owned by user 3
```

Para confirmar que o usuário 4 pode usar a chave pública (e todas as chaves públicas no HSM), use o parâmetro `-u` de [findKey](#).

A saída mostra que o usuário 4 pode usar tanto a chave pública (21) quanto a privada (20) no par de chaves. O Usuário 4 também pode usar todas as outras chaves públicas e quaisquer chaves privadas que ele tenha criado ou que tenha sido compartilhada com ele.

```
Command: findKey -u 4
```

```
Total number of keys present 8
```

```
number of keys matched from start index 0::7
```

```
11, 12, 262159, 262161, 262162, 19, 20, 21
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: obter o valor de autenticação de quorum (`m_value`) para uma chave

Esse exemplo mostra como obter o `m_value` para uma chave, ou seja, o número de usuários no quorum que devem aprovar todas as operações criptográficas que usam a chave.

Quando a autenticação de quorum está habilitada em uma chave, um quorum de usuários deve aprovar todas as operações criptográficas que usam essa chave. Para habilitar a autenticação de quorum e definir o tamanho do quorum, use o parâmetro `-m_value` ao criar a chave.

Esse comando usa [gen RSAKey Pair para criar um par](#) de chaves RSA que é compartilhado com o usuário 4. Ele usa o parâmetro `m_value` para habilitar a autenticação de quorum na chave privada no par e definir o tamanho do quorum para dois usuários. O número de usuários deve ser grande o suficiente para fornecer as aprovações necessárias.

A saída mostra que o comando criou a chave pública 27 e a chave privada 28.

```
Command: genRSAKeyPair -m 2048 -e 195193 -l rsa_mofn -id rsa_mv2 -u 4 -m_value 2

Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

Cfm3GenerateKeyPair:    public key handle: 27    private key handle: 28

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Esse comando usa `getKeyInfo` para obter informações sobre os usuários da chave privada. A saída mostra que a chave é de propriedade do usuário 3 e compartilhada com o usuário 4. Também mostra que um quorum de dois usuários deve aprovar todas as operações criptográficas que usam a chave.

```
Command: getKeyInfo -k 28

Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

Owned by user 3

also, shared to following 1 user(s):

    4

2 Users need to approve to use/manage this key
```

Parâmetros

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-k

Especifica o identificador da chave no HSM. Insira o identificador de uma chave que você possui ou compartilha. Esse parâmetro é obrigatório.

Para encontrar os identificadores de chave, use o comando [findKey](#).

Obrigatório: Sim

Tópicos relacionados

- [getKeyInfo](#) em cloudhsm_mgmt_util
- [listUsers](#)
- [findKey](#)
- [findAllKeys](#) em cloudhsm_mgmt_util

Exibir informações de ajuda para o AWS CloudHSM KMU

Use o help comando no AWS CloudHSM key_mgmt_util para exibir informações sobre todos os comandos key_mgmt_util disponíveis.

Antes de executar help, você deve [iniciar key_mgmt_util](#).

Sintaxe

```
help
```

Exemplo

Este exemplo mostra a saída do comando help.

Example

Command: **help**

Help Commands Available:

Syntax: <command> -h

Command	Description
=====	=====
exit	Exits this application
help	Displays this information
Configuration and Admin Commands	
getHSMInfo	Gets the HSM Information
getPartitionInfo	Gets the Partition Information
listUsers	Lists all users of a partition

loginStatus	Gets the Login Information
loginHSM	Login to the HSM
logoutHSM	Logout from the HSM

M of N commands

getToken	Initiate an MxN service and get Token
delToken	delete Token(s)
approveToken	Approves an MxN service
listTokens	List all Tokens in the current partition

Key Generation Commands

Asymmetric Keys:

genRSAKeyPair	Generates an RSA Key Pair
genDSAKeyPair	Generates a DSA Key Pair
genECCKeyPair	Generates an ECC Key Pair

Symmetric Keys:

genPBEKey	Generates a PBE DES3 key
genSymKey	Generates a Symmetric keys

Key Import/Export Commands

createPublicKey	Creates an RSA public key
importPubKey	Imports RSA/DSA/EC Public key
exportPubKey	Exports RSA/DSA/EC Public key
importPrivateKey	Imports RSA/DSA/EC private key
exportPrivateKey	Exports RSA/DSA/EC private key
imSymKey	Imports a Symmetric key
exSymKey	Exports a Symmetric key
wrapKey	Wraps a key from from HSM using the specified handle
unwrapKey	UnWraps a key into HSM using the specified handle

Key Management Commands

deleteKey	Delete Key
setAttribute	Sets an attribute of an object
getKeyInfo	Get Key Info about shared users/sessions
findKey	Find Key
findSingleKey	Find single Key
getAttribute	Reads an attribute from an object

Certificate Setup Commands

getCert	Gets Partition Certificates stored on HSM

Key Transfer Commands

<code>insertMaskedObject</code>	Inserts a masked object
<code>extractMaskedObject</code>	Extracts a masked object
Management Crypto Commands	
<code>sign</code>	Generates a signature
<code>verify</code>	Verifies a signature
<code>aesWrapUnwrap</code>	Does NIST AES Wrap/Unwrap
Helper Commands	
<code>Error2String</code>	Converts Error codes to Strings
<code>saveKeyHandleInFakePEMFormat</code>	Saves an RSA private key handle in fake PEM format
<code>getCaviumPrivKey</code>	Saves an RSA private key handle in fake PEM format
<code>IsValidKeyHandleFile</code>	Checks if private key file has an HSM key handle or a real key
<code>listAttributes</code>	List all attributes for <code>getAttribute</code>
<code>listECCCurveIds</code>	List HSM supported ECC CurveIds

Parâmetros

Não há parâmetros para esse comando.

Tópicos relacionados

- [loginHSM e logoutHSM](#)

Importar uma chave privada usando o AWS CloudHSM KMU

Use o `importPrivateKey` comando no AWS CloudHSM `key_mgmt_util` para importar uma chave privada assimétrica de um arquivo para um módulo de segurança de hardware (HSM). O HSM não permite a importação direta de chaves em texto não criptografado. O comando criptografa a chave privada usando uma chave de empacotamento AES que você especifica e desempacota a chave dentro do HSM. Se você estiver tentando associar uma AWS CloudHSM chave a um certificado, consulte [este tópico](#).

Note

Não é possível importar uma chave PEM protegida por senha usando uma chave simétrica ou privada.

Você deve especificar uma chave de empacotamento AES que tenha OBJ_ATTR_UNWRAP e OBJ_ATTR_ENCRYPT de valor de atributo 1. Para encontrar os atributos de uma chave, use o comando [getAttribute](#).

 Note

Esse comando não oferece a opção de marcar as chaves importadas como não exportáveis.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
importPrivateKey -h

importPrivateKey -l <label>
                  -f <key-file>
                  -w <wrapping-key-handle>
                  [-sess]
                  [-id <key-id>]
                  [-m_value <0...8>]
                  [min_srv <minimum-number-of-servers>]
                  [-timeout <number-of-seconds>]
                  [-u <user-ids>]
                  [-wk <wrapping-key-file>]
                  [-attest]
```

Exemplos

Este exemplo mostra como usar `importPrivateKey` para importar uma chave privada para um HSM.

Exemplo: importar uma chave privada

Esse comando importa a chave privada a partir de um arquivo chamado `rsa2048.key` com o rótulo `rsa2048-imported` e uma chave de encapsulamento com identificador 524299. Quando o comando for bem-sucedido, `importPrivateKey` retornará um identificador de chave para a chave importada e uma mensagem de êxito.

Command: **`importPrivateKey -f rsa2048.key -l rsa2048-imported -w 524299`**

```
BER encoded key length is 1216

Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Private Key Unwrapped.  Key Handle: 524301

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-l

Especifica o rótulo da chave privada definida pelo usuário.

Obrigatório: Sim

-f

Especifica o nome do arquivo da chave a ser importada.

Obrigatório: Sim

-w

Especifica o identificador de chave da chave de encapsulamento. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para determinar se uma chave pode ser usada como uma chave de encapsulamento, use [getAttribute](#) para obter o valor do atributo OBJ_ATTR_WRAP (262). Para criar uma chave de encapsulamento, use [genSymKey](#) a fim de criar uma chave AES (digite 31).

Se você usar o parâmetro `-wk` para especificar uma chave de desencapsulamento externa, a chave de encapsulamento `-w` será usada para encapsular, mas não para desencapsular, a chave durante a importação.

Obrigatório: Sim

-sess

Especifica a chave importada como uma chave de sessão.

Padrão: a chave importada é mantida como uma chave persistente (token) no cluster.

Obrigatório: não

-id

Especifica o ID da chave a ser importada.

Padrão: sem valor de ID.

Obrigatório: não

-m_value

Especifica o número de usuários que devem aprovar qualquer operação criptográfica que use a chave importada. Insira um valor de **0** a **8**.

Esse parâmetro é válido somente quando o parâmetro `-u` no comando compartilha a chave com usuários o suficiente para atender ao requisito de `m_value`.

Padrão: 0

Obrigatório: não

-min_srv

Especifica o número mínimo HSMs no qual a chave importada é sincronizada antes que o valor do `-timeout` parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de `min_srv` como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-timeout

Especifica o número de segundos de espera até que a chave seja sincronizada HSMs quando o `min-serv` parâmetro é incluído. Se nenhum número for especificado, a sondagem continuará para sempre.

Padrão: sem limite

Obrigatório: não

-u

Especifica a lista de usuários com os quais compartilhar a chave privada importada. Esse parâmetro dá permissão a outros usuários de criptografia do HSM (CUs) para usar a chave importada em operações criptográficas.

Insira uma lista separada por vírgulas de usuários do HSM IDs, como. `-u 5,6` Não inclua o ID do usuário atual do HSM. [Para encontrar o usuário IDs do HSM CUs no HSM, use ListUsers.](#)

Padrão: somente o usuário atual pode utilizar a chave importada.

Obrigatório: não

-wk

Especifica a chave a ser usada para encapsular a chave que está sendo importada. Insira o caminho e o nome de um arquivo que contém uma chave AES de texto simples.

Quando você inclui esse parâmetro, `importPrivateKey` usa a chave no arquivo `-wk` para encapsular a chave que está sendo importada. Ele também usa a chave especificada pelo parâmetro `-w` para desencapsulá-la.

Padrão: use a chave de encapsulamento especificada no parâmetro `-w` para encapsular e desencapsular.

Obrigatório: não

-attest

Executa uma verificação de declaração na resposta do firmware para garantir que o firmware no qual o cluster é executado não tenha sido comprometido.

Obrigatório: não

Tópicos relacionados

- [wrapKey](#)
- [unWrapKey](#)
- [genSymKey](#)
- [exportPrivateKey](#)

Importar uma chave pública usando o AWS CloudHSM KMU

Use o `importPubKey` comando no AWS CloudHSM `key_mgmt_util` para importar uma chave pública no formato PEM para um módulo de segurança de hardware (HSM). Você pode usá-lo para importar chaves públicas que foram geradas fora do HSM. Você também pode usar o comando para importar chaves que foram exportadas de um HSM, como as exportadas pelo comando [exportPubKey](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
importPubKey -h

importPubKey -l <label>
               -f <key-file>
               [-sess]
               [-id <key-id>]
               [min_srv <minimum-number-of-servers>]
               [-timeout <number-of-seconds>]
```

Exemplos

Este exemplo mostra como usar `importPubKey` para importar uma chave pública para um HSM.

Example: Import a Public Key

Esse comando importa uma chave pública de um arquivo chamado `public.pem` com o rótulo `importedPublicKey`. Quando o comando for bem-sucedido, `importPubKey` retornará um identificador de chave para a chave importada e uma mensagem de êxito.

```
Command: importPubKey -l importedPublicKey -f public.pem
```

```
Cfm3CreatePublicKey returned: 0x00 : HSM Return: SUCCESS
```

```
Public Key Handle: 262230
```

```
Cluster Error Status
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-l

Especifica o rótulo da chave pública definido pelo usuário.

Obrigatório: Sim

-f

Especifica o nome do arquivo da chave a ser importada.

Obrigatório: Sim

-sess

Designa a chave importada como uma chave de sessão.

Padrão: a chave importada é mantida como uma chave persistente (token) no cluster.

Obrigatório: não

-id

Especifica o ID da chave a ser importada.

Padrão: sem valor de ID.

Obrigatório: não

-min_srv

Especifica o número mínimo com o qual HSMs a chave importada é sincronizada antes que o valor do `-timeout` parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de `min_srv` como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-timeout

Especifica o número de segundos de espera até que a chave seja sincronizada HSMs quando o `min-serv` parâmetro é incluído. Se nenhum número for especificado, a sondagem continuará para sempre.

Padrão: sem limite

Obrigatório: não

Tópicos relacionados

- [exportPubKey](#)
- [Gerar chaves](#)

Importe uma chave simétrica de texto simples usando KMU AWS CloudHSM

Use o `imSymKey` comando na ferramenta AWS CloudHSM `key_mgmt_util` para importar uma cópia em texto simples de uma chave simétrica de um arquivo para o módulo de segurança de hardware (HSM). Você pode usá-lo para importar chaves geradas por qualquer método fora do HSM e chaves que foram exportadas de um HSM, como as chaves que o [exSymKey](#) comando grava em um arquivo.

Durante o processo de importação, `imSymKey` usa uma chave AES que você seleciona (a chave de encapsulamento) para encapsular (criptografar) e depois desencapsular (descriptografar) a chave a ser importada. No entanto, `imSymKey` funciona apenas em arquivos que contenham chaves em texto simples. Para exportar e importar chaves criptografadas, use o [WrapKey](#) e [unWrapKey](#) comandos.

Além disso, o comando `imSymKey` importa apenas chaves simétricas. Para importar chaves públicas, use [importPubKey](#). Para importar chaves privadas, use [importPrivateKey](#) ou [wrapKey](#).

Note

Não é possível importar uma chave PEM protegida por senha usando uma chave simétrica ou privada.

As chaves importadas funcionam muito como as chaves geradas no HSM. No entanto, o valor do [atributo OBJ_ATTR_LOCAL](#) é zero, o que indica que não foi gerado localmente. Você pode usar o comando a seguir para compartilhar uma chave simétrica ao importá-la. Você pode usar o comando `shareKey` em [cloudhsm_mgmt_util](#) para compartilhar a chave após ela ser importada.

```
imSymKey -l aesShared -t 31 -f kms.key -w 3296 -u 5
```

Depois de importar uma chave, certifique-se de marcar ou excluir o arquivo de chave. Esse comando não impede que você importe o mesmo material de chave várias vezes. O resultado, várias chaves com identificadores de chave distintos e o mesmo material de chave, dificultam o rastreamento do uso do material de chave e impedem que ela exceda seus limites criptográficos.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
imSymKey -h

imSymKey -f <key-file>
          -w <wrapping-key-handle>
          -t <key-type>
          -l <label>
          [-id <key-ID>]
          [-sess]
          [-wk <wrapping-key-file> ]
```



```
[-attest]
[-min_srv <minimum-number-of-servers>]
[-timeout <number-of-seconds> ]
[-u <user-ids>]
```

Exemplos

Esses exemplos mostram como usar `imSymKey` para importar chaves simétricas para o seu HSMs.

Example: Importar uma chave simétrica AES

Este exemplo usa `imSymKey` para importar uma chave simétrica AES para o HSMs

O primeiro comando usa o OpenSSL para gerar uma chave AES simétrica aleatória de 256 bits. Ele salva a chave no arquivo `aes256.key`.

```
$ openssl rand -out aes256.key 32
```

O segundo comando é usado `imSymKey` para importar a chave AES do `aes256.key` arquivo para HSMs o. Ele usa a chave 20, uma chave AES no HSM, como a chave de encapsulamento e especifica um rótulo de `imported`. Ao contrário do ID, o rótulo não precisa ser exclusivo no cluster. O valor do parâmetro `-t` (tipo) é 31, que representa o AES.

A saída mostra que a chave no arquivo foi encapsulada e desencapsulada, e depois importada para o HSM, onde recebeu o identificador de chave 262180.

```
Command: imSymKey -f aes256.key -w 20 -t 31 -l imported
```

```
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Unwrapped. Key Handle: 262180
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

O próximo comando usa [getAttribute](#) para obter o atributo OBJ_ATTR_LOCAL ([atributo 355](#)) da chave recém-importada e o grava no arquivo `attr_262180`.

```
Command: getAttribute -o 262180 -a 355 -out attributes/attr_262180  
Attributes dumped into attributes/attr_262180_imported file
```

```
Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS
```

Ao examinar o arquivo de atributo, você pode ver que o valor do atributo OBJ_ATTR_LOCAL é zero, o que indica que o material de chave não foi gerado no HSM.

```
$ cat attributes/attr_262180_local  
OBJ_ATTR_LOCAL  
0x00000000
```

Example: mover uma chave simétrica entre clusters

Este exemplo mostra como usar [exSymKey](#) e `imSymKey` para mover uma chave AES de texto simples entre clusters. Você pode usar um processo como esse para criar um encapsulamento AES que exista nos HSMs dois clusters. Depois que a chave de empacotamento compartilhada estiver em vigor, você poderá usar [WrapKey](#) [unWrapKey](#) e mover chaves criptografadas entre os clusters.

O usuário da UC que executa essa operação deve ter permissão para fazer login no em ambos os clusters. HSMs

O primeiro comando usa [exSymKey](#) para exportar a chave 14, uma chave AES de 32 bits, do cluster 1 para o arquivo `aes.key`. Ele usa a chave 6, uma chave AES HSMs no cluster 1, como chave de empacotamento.

```
Command: exSymKey -k 14 -w 6 -out aes.key
```

```
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

```
Wrapped Symmetric Key written to file "aes.key"
```

Em seguida, o usuário faz login no `key_mgmt_util` no cluster 2 e executa um `imSymKey` comando para importar a chave no `aes.key` arquivo para o cluster 2. HSMs Esse comando usa a chave 252152, uma chave AES HSMs no cluster 2, como chave de encapsulamento.

Como as chaves de encapsulamento usadas por [exSymKey](#) e `imSymKey` encapsulam e imediatamente desencapsulam as chaves de destino, as chaves de encapsulamento nos diferentes clusters não precisam ser iguais.

A saída mostra que a chave foi importada com sucesso no cluster 2 e recebeu um identificador de chave de 21.

```
Command: imSymKey -f aes.key -w 262152 -t 31 -l xcluster

Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Unwrapped. Key Handle: 21

Cluster Error Status
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Para provar que a chave 14 no cluster 1 e a chave 21 no cluster 2 possuem o mesmo material de chave, obtenha o valor de verificação de chave (KCV) de cada uma. Se os valores do KCV forem iguais, o material da chave é o mesmo.

O seguinte comando usa [getAttribute](#) no cluster 1 para gravar o valor do atributo KCV (atributo 371) da chave 14 no arquivo `attr_14_kcv`. Em seguida, ele usa um comando `cat` para obter o conteúdo do arquivo `attr_14_kcv`.

```
Command: getAttribute -o 14 -a 371 -out attr_14_kcv
Attributes dumped into attr_14_kcv file

$ cat attr_14_kcv
OBJ_ATTR_KCV
0xc33cbd
```

Esse comando semelhante usa [getAttribute](#) no cluster 2 para gravar o valor do atributo KCV (atributo 371) da chave 21 no arquivo `attr_21_kcv`. Em seguida, ele usa um comando `cat` para obter o conteúdo do arquivo `attr_21_kcv`.

```
Command: getAttribute -o 21 -a 371 -out attr_21_kcv  
Attributes dumped into attr_21_kcv file  
  
$ cat attr_21_kcv  
OBJ_ATTR_KCV  
0xc33cbd
```

A saída mostra que os valores KCV das duas chaves são os mesmos, o que prova que o material de chave é o mesmo.

Como o mesmo material HSMs de chave existe nos dois clusters, agora você pode compartilhar chaves criptografadas entre os clusters sem nunca expor a chave de texto simples. Por exemplo, você pode usar o comando `wrapKey` com a chave de encapsulamento 14 para exportar uma chave criptografada do cluster 1 e, em seguida, usar `unWrapKey` com a chave de encapsulamento 21 para importar a chave criptografada para o cluster 2.

Example: Importar uma chave de sessão

Esse comando usa os parâmetros `-sess` de `imSymKey` para importar uma chave Triple DES de 192 bits que é válida apenas na sessão atual.

O comando usa o parâmetro `-f` para especificar o arquivo que contém a chave a ser importada, o parâmetro `-t` para especificar o tipo de chave e o parâmetro `-w` para especificar a chave de encapsulamento. Ele usa o parâmetro `-l` para especificar um rótulo que categoriza a chave e o parâmetro `-id` para criar um identificador amigável, mas exclusivo, para ela. Ele também usa o parâmetro `-attest` para verificar o firmware que está importando a chave.

A saída mostra que a chave foi encapsulada e desencapsulada com sucesso, importada para o HSM e recebeu o identificador de chave 37. Além disso, a verificação de atestado foi aprovada, o que indica que o firmware não foi adulterado.

```
Command: imSymKey -f 3des192.key -w 6 -t 21 -l temp -id test01 -sess -attest  
  
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS  
  
Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS  
  
Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS  
  
Symmetric Key Unwrapped. Key Handle: 37
```

```
Attestation Check : [PASS]
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Em seguida, você pode usar os comandos [getAttribute](#) ou [findKey](#) para verificar os atributos da chave recém-importada. O comando a seguir usa `findKey` para verificar se a chave 37 possui o tipo, o rótulo e o ID especificados pelo comando e se ela é uma chave de sessão. Como mostrado na linha 5 da saída, `findKey` informa que a única chave que corresponde a todos os atributos é a chave 37.

```
Command: findKey -t 21 -l temp -id test01 -sess 1
```

```
Total number of keys present 1
```

```
number of keys matched from start index 0::0
37
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

-attest

Executa uma verificação de integridade que verifica se o firmware no qual o cluster é executado não foi adulterado.

Padrão: sem verificação de atestado.

Obrigatório: não

-f

Especifica o arquivo que contém essa chave a ser importada.

O arquivo deve conter uma cópia em texto simples de uma chave AES ou DES tripla do tamanho especificado. RC4 e as chaves DES não são válidas no modo FIPS HSMs.

- AES: 16, 24 ou 32 bytes

- Triple DES (3DES): 24 bytes

Obrigatório: Sim

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-id

Especifica o identificador da chave definida pelo usuário. Digite uma string exclusiva no cluster. O padrão é uma string vazia.

Padrão: sem valor de ID.

Obrigatório: não

-l

Especifica o rótulo da chave privada definida pelo usuário para a chave. Digite uma string.

É possível usar qualquer frase que ajude a identificar a chave. Como o rótulo não precisa ser exclusivo, é possível usá-lo para agrupar e categorizar chaves.

Obrigatório: Sim

-min_srv

Especifica o número mínimo HSMs no qual a chave é sincronizada antes que o valor do -timeout parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de min_srv como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-sess

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [setAttribute](#).

Padrão: a chave é persistente.

Obrigatório: não

-timeout

Especifica por quanto tempo (em segundos) o comando espera que uma chave seja sincronizada com o número HSMs especificado pelo parâmetro. `min_srv`

Este parâmetro é válido somente quando o parâmetro `min_srv` também é usado no comando.

Padrão: sem limite de tempo. O comando espera indefinidamente e retorna somente quando a chave é sincronizada com o número mínimo de servidores.

Obrigatório: não

-t

Especifica o tipo da chave simétrica. Insira a constante que representa o tipo de chave. Por exemplo, para criar uma chave AES, insira `-t 31`.

Valores válidos:

- 21: [Triple DES \(3DES\)](#).
- 31: [AES](#)

Obrigatório: Sim

-u

Compartilha a chave que você está importando com usuários especificados. Esse parâmetro dá permissão a outros usuários de criptografia do HSM (CUs) para usar essa chave em operações criptográficas.

Digite um ID ou uma lista separada por vírgulas do usuário do HSM IDs, como `-u. 5, 6` Não inclua o ID do usuário atual do HSM. Para encontrar um ID, você pode usar o comando [listUsers](#) na

ferramenta de linha de comando `cloudhsm_mgmt_util` ou o comando [listUsers](#) na ferramenta de linha de comando `key_mgmt_util`.

Obrigatório: não

-w

Especifica o identificador de chave da chave de encapsulamento. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Uma chave de encapsulamento é uma chave no HSM que é usada para criptografar (encapsular) e depois descriptografar (desencapsular) a chave durante o processo de importação. Somente as chaves AES podem ser usadas como chaves de encapsulamento.

Você pode usar qualquer chave do AES (de qualquer tamanho) como uma chave de encapsulamento. Como a chave de encapsulamento encapsula e depois desencapsula imediatamente a chave de destino, você pode usar como chave AES somente de sessão como uma chave de encapsulamento. Para determinar se uma chave pode ser usada como uma chave de encapsulamento, use [getAttribute](#) para obter o valor do atributo OBJ_ATTR_WRAP (262). Para criar uma chave de encapsulamento, use [genSymKey](#) a fim de criar uma chave AES (digite 31).

Se você usar o parâmetro `-wk` para especificar uma chave de encapsulamento externa, a chave de encapsulamento `-w` será usada para desencapsular, mas não para encapsular, a chave que está sendo importada.

 Note

A chave 4 é uma chave interna sem suporte. Recomendamos usar uma chave AES criada e gerenciada por você como a chave de encapsulamento.

Obrigatório: Sim

-wk

Use a chave AES no arquivo especificado para encapsular a chave que está sendo importada. Insira o caminho e o nome de um arquivo que contém uma chave AES de texto simples.

Durante a inclusão desse parâmetro, `imSymKey` usa a chave no arquivo `-wk` para encapsular a chave que está sendo importada e usa a chave no HSM que é especificada pelo parâmetro `-w` para desencapsulá-la. Os valores dos parâmetros `-w` e `-wk` devem ser resolvidos para a mesma chave de texto simples.

Padrão: use a chave de encapsulamento no HSM para desencapsular.

Obrigatório: não

Tópicos relacionados

- [genSymKey](#)
- [exSymKey](#)
- [wrapKey](#)
- [unWrapKey](#)
- [exportPrivateKey](#)
- [exportPubKey](#)

Inserir um objeto mascarado usando o AWS CloudHSM KMU

Use o `insertMaskedObject` comando no AWS CloudHSM `key_mgmt_util` para inserir um objeto mascarado de um arquivo em um módulo de segurança de hardware (HSM) designado.

Objetos mascarados são objetos clonados que são extraídos de um HSM usando o comando [extractMaskedObject](#). Eles só podem ser usados após inseri-los novamente no cluster original. Você só pode inserir um objeto mascarado no mesmo cluster do qual ele foi gerado ou em um clone desse cluster. Isso inclui quaisquer versões clonadas do cluster original geradas ao [copiar um backup entre regiões](#) e [usar esse backup para criar um novo cluster](#).

Objetos mascarados são uma maneira eficiente de descarregar e sincronizar chaves, incluindo chaves nonextractable (isto é, chaves que têm um `OBJ_ATTR_EXTRACTABLE` valor de 0). [Dessa forma, as chaves podem ser sincronizadas com segurança entre clusters relacionados em diferentes regiões sem a necessidade de atualizar o AWS CloudHSM arquivo de configuração.](#)

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
insertMaskedObject -h

insertMaskedObject -f <filename>
                    [-min_srv <minimum-number-of-servers>]
                    [-timeout <number-of-seconds>]
```

Exemplos

Este exemplo mostra como usar `insertMaskedObject` para inserir um arquivo de objeto mascarado em um HSM.

Example: Insert a Masked Object

Esse comando insere um objeto mascarado em um HSM de um arquivo denominado `maskedObj`. Quando o comando for bem-sucedido, `insertMaskedObject` retornará um identificador de chave para a chave descryptografada do objeto mascarado e uma mensagem de êxito.

```
Command: insertMaskedObject -f maskedObj
```

```
Cfm3InsertMaskedObject returned: 0x00 : HSM Return: SUCCESS  
New Key Handle: 262433
```

```
Cluster Error Status
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-f

Especifica o nome do arquivo do objeto mascarado a ser inserido.

Obrigatório: Sim

-min_srv

Especifica o número mínimo de servidores nos quais o objeto mascarado inserido é sincronizado antes da expiração do valor do parâmetro `-timeout`. Se o objeto não for sincronizado com o número especificado de servidores na hora alocada, ela não será inserido.

Padrão: 1

Obrigatório: não

-timeout

Especifica o número de segundos a aguardar a sincronização da chave entre os servidores quando o parâmetro `min-serv` é incluído. Se nenhum número for especificado, a sondagem continuará para sempre.

Padrão: sem limite

Obrigatório: não

Tópicos relacionados

- [extractMaskedObject](#)
- [syncKey](#)
- [Copiar um backup entre regiões](#)
- [Criando um AWS CloudHSM cluster a partir de um backup anterior](#)

Valide o arquivo de chave usando AWS CloudHSM o KMU

Use o `IsValidKeyHandlefile` comando no AWS CloudHSM `key_mgmt_util` para descobrir se um arquivo de chave contém uma chave privada real ou uma chave RSA PEM falsa. Um arquivo PEM falso não contém o material da chave privada real, mas, em vez disso, faz referência à chave privada no HSM. Esse arquivo pode ser usado para estabelecer o SSL/TLS descarregamento do seu servidor web para o AWS CloudHSM. Para obter mais informações, consulte [SSL/TLS Offload on Linux using Tomcat](#) ou [SSL/TLS Offload on Linux using NGINX or Apache](#).

Note

`IsValidKeyHandlefile` funciona apenas para chaves RSA.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
IsValidKeyHandlefile -h
```

```
IsValidKeyHandlefile -f <rsa-private-key-file>
```

Exemplos

Esses exemplos mostram como usar `IsValidKeyHandlefile` para determinar se um determinado arquivo de chave contém o material de chave real ou o material de chave PEM falsa.

Example: Validate a Real Private Key

Esse comando confirma que o arquivo chamado `privateKey.pem` contém o material de chave real.

```
Command: IsValidKeyHandlefile -f privateKey.pem
```

```
Input key file has real private key
```

Example: Invalidate a Fake PEM Key

Esse comando confirma que o arquivo chamado `caviumKey.pem` contém o material de chave PEM falsa feito no identificador de chave 15.

```
Command: IsValidKeyHandlefile -f caviumKey.pem
```

```
Input file has invalid key handle: 15
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-h

Exibe a ajuda da linha de comando para o comando.

Obrigatório: Sim

-f

Especifica o nome do arquivo de chave privada RSA a ser verificado para ver se há material de chave válido.

Obrigatório: Sim

Tópicos relacionados

- [getCaviumPrivChave](#)
- [Descarregamento de SSL/TLS no Linux usando Tomcat](#)
- [Descarregamento de SSL/TLS no Linux usando NGINX ou Apache](#)

Listar os atributos de uma AWS CloudHSM chave usando KMU

Use o listAttributes comando no AWS CloudHSM key_mgmt_util para listar os atributos de uma AWS CloudHSM chave e as constantes que os representam. Você usa essas constantes para identificar os atributos nos comandos [getAttribute](#) e [setAttribute](#). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

Este comando não possui parâmetros.

```
listAttributes
```

Exemplo

Este comando lista os principais atributos que você pode obter e alterar em key_mgmt_util, bem como as constantes que os representam. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Para representar todos os atributos no comando [getAttribute](#) em key_mgmt_util, use 512.

Command: **listAttributes**

Following are the possible attribute values for getAttributes:

OBJ_ATTR_CLASS	= 0
OBJ_ATTR_TOKEN	= 1
OBJ_ATTR_PRIVATE	= 2
OBJ_ATTR_LABEL	= 3
OBJ_ATTR_KEY_TYPE	= 256
OBJ_ATTR_ENCRYPT	= 260

OBJ_ATTR_DECRYPT	= 261
OBJ_ATTR_WRAP	= 262
OBJ_ATTR_UNWRAP	= 263
OBJ_ATTR_SIGN	= 264
OBJ_ATTR_VERIFY	= 266
OBJ_ATTR_LOCAL	= 355
OBJ_ATTR_MODULUS	= 288
OBJ_ATTR_MODULUS_BITS	= 289
OBJ_ATTR_PUBLIC_EXPONENT	= 290
OBJ_ATTR_VALUE_LEN	= 353
OBJ_ATTR_EXTRACTABLE	= 354
OBJ_ATTR_KCV	= 371

Tópicos relacionados

- [listAttributes](#) em cloudhsm_mgmt_util
- [getAttribute](#)
- [setAttribute](#)
- [Referência de atributos de chave](#)

Listar todos os AWS CloudHSM usuários usando KMU

Use o listUsers comando no AWS CloudHSM key_mgmt_util para colocar os usuários nos módulos de segurança de hardware (HSM), junto com o tipo de usuário e outros atributos.

Em key_mgmt_util, ListUsers retorna uma saída que representa HSMs tudo no cluster, mesmo que não sejam consistentes. Para obter informações sobre os usuários em cada HSM, use o comando [listUsers](#) em cloudhsm_mgmt_util.

Os comandos do usuário em key_mgmt_util listUsers e [getKeyInfo](#), são comandos somente para leitura que os usuários criptográficos (CUs) têm permissão para executar. Os comandos de gerenciamento de usuários restantes fazem parte da cloudhsm_mgmt_util. Eles são administrados por responsáveis pela criptografia (CO) que têm permissões de gerenciamento de usuários.

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
listUsers
```

```
listUsers -h
```

Exemplo

Esse comando lista os usuários HSMs do cluster e seus atributos. Você pode usar o User ID atributo para identificar usuários em outros comandos, como [FindKey](#), [getAttribute](#) e [getKeyInfo](#).

Command: `listUsers`

Number Of Users found 4

Index	User ID	User Type	User Name	MofnPubKey
0	1	PCO	admin	NO
0	2	AU	app_user	NO
0	3	CU	alice	YES
0	4	CU	bob	NO
0	5	CU	trent	YES

Cfm3ListUsers returned: 0x00 : HSM Return: SUCCESS

A saída inclui os seguintes atributos do usuário:

- User ID: identifica o usuário em comandos da `key_mgmt_util` e do [cloudhsm_mgmt_util](#).
- [User type](#): determina as operações que o usuário pode executar no HSM.
- User Name: exibe o nome amigável definido pelo usuário para o usuário.
- MofnPubKey: indica se o usuário registrou um par de chaves para assinar tokens de [autenticação de quórum](#).
- LoginFailureCnt: indica o número de vezes que o usuário fez login sem sucesso.
- 2FA: indica que o usuário habilitou a autenticação multifator.

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

Tópicos relacionados

- [ListUsers](#) em cloudhsm_mgmt_util
- [findKey](#)
- [getAttribute](#)
- [getKeyInfo](#)

Faça login e saia de um HSM usando AWS CloudHSM KMU

Use os comandos loginHSM e logoutHSM no key_mgmt_util do AWS CloudHSM para fazer login e logout dos módulos de segurança de hardware (HSM) em um cluster. Depois de fazer login no HSMs, você pode usar key_mgmt_util para realizar uma variedade de operações de gerenciamento de chaves, incluindo geração, sincronização e agrupamento de chaves públicas e privadas.

Antes de executar qualquer comando key_mgmt_util, você deve [iniciar key_mgmt_util](#). [Para gerenciar chaves com key_mgmt_util, você deve fazer login no HSMs como usuário criptográfico \(CU\)](#).

Note

Se você exceder cinco tentativas incorretas de login, sua conta será bloqueada. Se você criou seu cluster antes de fevereiro de 2018, sua conta será bloqueada após 20 tentativas incorretas de login. Para desbloquear a conta, um responsável pela criptografia (CO) precisará redefinir sua senha usando o comando [changePswd](#) no cloudhsm_mgmt_util. Se você tiver mais de um HSM no cluster, pode ser que mais tentativas de login sejam permitidas antes de a sua conta ser bloqueada. Isso ocorre porque o cliente CloudHSM equilibra a carga em vários HSMs. Sendo assim, pode ser que as tentativas de login não comecem sempre no mesmo HSM. Se você estiver testando essa funcionalidade, recomendamos fazer isso em um cluster com apenas um HSM ativo.

Sintaxe

```
loginHSM -h

loginHSM -u <user type>
          { -p | -hpswd } <password>
          -s <username>
```

Exemplo

Este exemplo mostra como fazer login e sair do HSMs em um cluster com os `logoutHSM` comandos `loginHSM` e.

Example: Faça login no HSMs

Esse comando conecta você HSMs como usuário criptográfico (CU) com o nome de usuário `example_user` e a senha `aws`. A saída mostra que você fez login em tudo HSMs no cluster.

```
Command: loginHSM -u CU -s example_user -p aws

Cfm3LoginHSM returned: 0x00 : HSM Return: SUCCESS

Cluster Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Faça login com uma senha oculta

Esse comando é o mesmo do exemplo acima, exceto que, desta vez, você especifica que o sistema deve ocultar a senha.

```
Command: loginHSM -u CU -s example_user -hpswd
```

O sistema solicita que você forneça sua senha. Você insere a senha, o sistema oculta a senha e a saída mostra que o comando foi bem-sucedido e que você se conectou ao HSMs.

```
Enter password:

Cfm3LoginHSM returned: 0x00 : HSM Return: SUCCESS
```

Cluster Status

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Command:

Example: Sair do HSMs

Esse comando desconecta você do HSMs. A saída mostra que você se desconectou de tudo HSMs no cluster.

Command: **logoutHSM**

```
Cfm3LogoutHSM returned: 0x00 : HSM Return: SUCCESS
```

Cluster Status

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

-h

Exibe a ajuda referente a esse comando.

-u

Especifica o tipo de usuário de login. Para usar a `key_mgmt_util`, você deve fazer login como um CU.

Obrigatório: Sim

-s

Especifica o nome de usuário de login.

Obrigatório: Sim

{-p | -hpswd}

Especifique a senha de login com `-p`. A senha aparece em texto sem formatação quando você a digita. Para ocultar sua senha, use o `-hpswd` parâmetro opcional no lugar da senha `-p` e siga as instruções.

Obrigatório: Sim

Tópicos relacionados

- [exit](#)

Defina os atributos das AWS CloudHSM chaves usando o KMU

Use o `setAttribute` comando no AWS CloudHSM `key_mgmt_util` para converter uma chave válida somente na sessão atual em uma chave persistente que existe até você excluí-la. Ele faz isso alterando o valor do atributo de token da chave (`OBJ_ATTR_TOKEN`) de `false (0)` para `true (1)`. Você só pode alterar os atributos das chaves que possui.

Você também pode usar o comando `setAttribute` em `cloudhsm_mgmt_util` para alterar os atributos de rótulos, encapsulamento, desencapsulamento, criptografia e descriptografia.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
setAttribute -h

setAttribute -o <object handle>
               -a 1
```

Exemplo

Este exemplo mostra como converter uma chave de sessão em uma chave persistente.

O primeiro comando usa o `-sess` parâmetro de [genSymKey](#) para criar uma chave AES de 192 bits que é válida somente na sessão atual. A saída mostra que o identificador da nova chave de sessão é 262154.

```
Command: genSymKey -t 31 -s 24 -l tmpAES -sess
```

```
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS
```

```
Symmetric Key Created. Key Handle: 262154
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Esse comando usa [findKey](#) para encontrar as chaves de sessão na sessão atual. A saída verifica que a chave 262154 é uma chave de sessão.

```
Command: findKey -sess 1
```

```
Total number of keys present 1
```

```
number of keys matched from start index 0::0  
262154
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Esse comando usa `setAttribute` para converter a chave 262154 de uma chave de sessão em uma chave persistente. Para fazer isso, ele muda o valor do atributo de token (`OBJ_ATTR_TOKEN`) da chave de 0 (false) para 1 (true). Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

O comando usa o parâmetro `-o` para especificar o identificador de chave (262154) e o parâmetro `-a` para especificar a constante que representa o atributo de token (1). Quando você executa o comando, ele solicita um valor para o atributo de token. O único valor válido é 1 (true); o valor de uma chave persistente.

```
Command: setAttribute -o 262154 -a 1
```

```
This attribute is defined as a boolean value.
```

```
Enter the boolean attribute value (0 or 1):1
```

```
Cfm3SetAttribute returned: 0x00 : HSM Return: SUCCESS
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Para confirmar que a chave 262154 agora é persistente, esse comando usa `findKey` para procurar chaves de sessão (`-sess 1`) e chaves persistentes (`-sess 0`). Dessa vez, o comando não encontra nenhuma chave de sessão, mas retorna 262154 na lista de chaves persistentes.

```
Command: findKey -sess 1
```

```
Total number of keys present 0
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

```
Command: findKey -sess 0
```

```
Total number of keys present 5
```

```
number of keys matched from start index 0::4
```

```
6, 7, 524296, 9, 262154
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

`-h`

Exibe a ajuda referente ao comando.

Obrigatório: Sim

`-o`

Especifica o identificador da chave de destino. Você pode especificar apenas uma chave em cada comando. Para obter o identificador de chave de uma chave, use [findKey](#).

Obrigatório: Sim

-a

Especifica a constante que representa o atributo que você deseja alterar. O único valor válido é 1, que representa o atributo de token, OBJ_ATTR_TOKEN.

Para obter os atributos e seus valores inteiros, use [listAttributes](#).

Obrigatório: Sim

Tópicos relacionados

- [setAttribute](#) em cloudhsm_mgmt_util
- [getAttribute](#)
- [listAttributes](#)
- [Referência de atributos de chave](#)

Gere uma assinatura usando o AWS CloudHSM KMU

Use o sign comando no AWS CloudHSM key_mgmt_util para usar uma chave privada escolhida para gerar uma assinatura para um arquivo.

Para usar sign, primeiro você deve ter uma chave privada no HSM. Você pode gerar uma chave privada com os comandos [genSymKey](#), [genRSAKeyPair](#) ou [genECCKeyPair](#). Você também pode importar uma com o comando [importPrivateKey](#). Para mais informações, consulte [Gerar chaves](#).

O comando sign usa um mecanismo de assinatura designada pelo usuário, representado por um número inteiro, para assinar um arquivo de mensagem. Para obter uma lista de possíveis mecanismos de assinatura, consulte [Parâmetros](#).

Antes de executar um comando key_mgmt_util, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
sign -h

sign -f <file name>
      -k <private key handle>
      -m <signature mechanism>
```

```
-out <signed file name>
```

Exemplo

Este exemplo mostra como usar sign para assinar um arquivo.

Example: Sign a file

Esse comando assina um arquivo chamado messageFile com uma chave privada com identificador 266309. Ele usa o mecanismo de assinatura SHA256_RSA_PKCS (1) e salva o arquivo assinado resultante como signedFile.

```
Command: sign -f messageFile -k 266309 -m 1 -out signedFile
```

```
Cfm3Sign returned: 0x00 : HSM Return: SUCCESS
```

```
signature is written to file signedFile
```

```
Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-f

O nome do arquivo a ser assinado.

Obrigatório: Sim

-k

O identificador da chave privada a ser usada para assinatura.

Obrigatório: Sim

-m

Um inteiro que representa o mecanismo de assinatura a ser usado para assinatura. Os mecanismos possíveis correspondem aos seguintes números inteiros:

Mecanismo de assinatura	Número inteiro correspondente
SHA1_RSA_PKCS	0
SHA256_RSA_PKCS	1
SHA384_RSA_PKCS	2
SHA512_RSA_PKCS	3
SHA224_RSA_PKCS	4
SHA1_RSA_PKCS_PSS	5
SHA256_RSA_PKCS_PSS	6
SHA384_RSA_PKCS_PSS	7
SHA512_RSA_PKCS_PSS	8
SHA224_RSA_PKCS_PSS	9
ECDSA_SHA1	15
ECDSA_SHA224	16
ECDSA_SHA256	17
ECDSA_SHA384	18
ECDSA_SHA512	19

Obrigatório: Sim

-out

O nome do arquivo no qual o arquivo assinado será salvo.

Obrigatório: Sim

Tópicos relacionados

- [verify](#)
- [importPrivateKey](#)
- [2 gênios RSAKey Pair](#)
- [2 gênios ECCKey Pair](#)
- [genSymKey](#)
- [Gerar chaves](#)

Desembrulhe uma AWS CloudHSM chave usando KMU

Use o `unWrapKey` comando na ferramenta AWS CloudHSM `key_mgmt_util` para importar uma chave simétrica ou privada encapsulada (criptografada) de um arquivo para o HSM. Ele é projetado para importar chaves criptografadas que foram encapsuladas pelo comando [wrapKey](#) em `key_mgmt_util`, mas também pode ser usado para desencapsular chaves que foram encapsuladas com outras ferramentas. No entanto, nessas situações, recomendamos usar as bibliotecas de software [PKCS#11](#) ou [JCE](#) para desencapsular a chave.

As chaves importadas funcionam como as chaves geradas por AWS CloudHSM. No entanto, o valor de seu [atributo OBJ_ATTR_LOCAL](#) é zero, o que indica que não foi gerada localmente.

Depois de importar uma chave, certifique-se de marcar ou excluir o arquivo de chave. Esse comando não impede que você importe o mesmo material de chave várias vezes. O resultado, várias chaves com identificadores de chave distintos e o mesmo material de chave, dificultam o rastreamento do uso do material de chave e impedem que elas excedam seus limites de criptografia.

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
unWrapKey -h

unWrapKey -f <key-file-name>
           -w <wrapping-key-handle>
           [-sess]
           [-min_srv <minimum-number-of-HSMs>]
           [-timeout <number-of-seconds>]
```

```

[-aad <additional authenticated data filename>]
[-tag_size <tag size>]
[-iv_file <IV file>]
[-attest]
[-m <wrapping-mechanism>]
[-t <hash-type>]
[-nex]
[-u <user id list>]
[-m_value <number of users needed for approval>]
[-noheader]
[-l <key-label>]
[-id <key-id>]
[-kt <key-type>]
[-kc <key-class>]
[-i <unwrapping-IV>]

```

Exemplo

Esses exemplos mostram como usar `unWrapKey` para importar uma chave encapsulada de um arquivo para HSMs o. No primeiro exemplo, desencapsulamos uma chave que foi encapsulada com o comando [wrapKey](#) `key_mgmt_util` e que, portanto, tem um cabeçalho. No segundo exemplo, desencapsulamos uma chave que foi encapsulada fora da `key_mgmt_util` e que, portanto, não tem um cabeçalho.

Example: desencapsular uma chave (com cabeçalho)

Esse comando importa uma cópia encapsulada de uma chave simétrica 3DES em um HSM. A chave é desencapsulada com uma chave AES com rótulo 6, que é criptograficamente idêntica a que foi usada para encapsular a chave 3DES. A saída mostra que a chave no arquivo foi desencapsulada e importada, e que o identificador da chave importada é 29.

Command: **unWrapKey -f 3DES.key -w 6 -m 4**

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Key Unwrapped. Key Handle: 29

Cluster Error Status

Node id 1 and err state 0x00000000 : HSM Return: SUCCESS

Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

Example: desencapsular uma chave (sem cabeçalho)

Esse comando importa uma cópia encapsulada de uma chave simétrica 3DES em um HSM. A chave é desencapsulada com uma chave AES com rótulo 6, que é criptograficamente idêntica a que foi usada para encapsular a chave 3DES. Como essa chave 3DES não foi encapsulada com `key_mgmt_util`, o parâmetro `noheader` é especificado, juntamente com seus parâmetros de acompanhamento necessários: um rótulo de chave (`unwrapped3DES`), uma classe de chave (4) e um tipo de chave (21). A saída mostra que a chave no arquivo foi desencapsulada e importada, e que o identificador da chave importada é 8.

```
Command: unWrapKey -f 3DES.key -w 6 -noheader -l unwrapped3DES -kc 4 -kt 21 -m 4
```

```
Cfm3CreateUnwrapTemplate2 returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm2UnWrapWithTemplate3 returned: 0x00 : HSM Return: SUCCESS
```

```
Key Unwrapped. Key Handle: 8
```

```
Cluster Error Status
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-f

O caminho e o nome do arquivo que contém a chave encapsulada.

Obrigatório: Sim

-w

Especifica a chave de empacotamento. Insira o identificador de uma chave AES ou chave RSA no HSM. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para criar uma chave de agrupamento, use [genSymKey](#) para gerar uma chave AES (tipo 31) ou [gen RSAKey Pair para gerar um par](#) de chaves RSA (tipo 0). Se você estiver usando um par de

chaves RSA, certifique-se de encapsular a chave com uma das chaves e desencapsular com a outra. Para determinar se uma chave pode ser usada como uma chave de empacotamento, use [getAttribute](#) para obter o valor do atributo OBJ_ATTR_WRAP, que é representado pela constante 262.

Obrigatório: Sim

-sess

Cria uma chave que existe apenas na sessão atual. A chave não pode ser recuperada após o término da sessão.

Use esse parâmetro quando precisar de uma chave apenas brevemente, como uma chave de encapsulamento que criptografa e, em seguida, descriptografa rapidamente outra chave. Não use uma chave de sessão para criptografar dados que você talvez precise descriptografar após o término da sessão.

Para transformar uma chave de sessão em uma chave persistente (token), use [setAttribute](#).

Padrão: a chave é persistente.

Obrigatório: não

-min_srv

Especifica o número mínimo HSMs no qual a chave é sincronizada antes que o valor do -timeout parâmetro expire. Se a chave não for sincronizada com o número especificado de servidores na hora alocada, ela não será criada.

AWS CloudHSM sincroniza automaticamente todas as chaves com cada HSM no cluster. Para acelerar seu processo, defina o valor de min_srv como menor que o número de HSMs no cluster e defina um valor de tempo limite baixo. No entanto, algumas solicitações talvez não gerem uma chave.

Padrão: 1

Obrigatório: não

-timeout

Especifica por quanto tempo (em segundos) o comando espera que uma chave seja sincronizada com o número HSMs especificado pelo parâmetro. min_srv

Este parâmetro é válido somente quando o parâmetro `min_srv` também é usado no comando.

Padrão: sem limite de tempo. O comando espera indefinidamente e retorna somente quando a chave é sincronizada com o número mínimo de servidores.

Obrigatório: não

-attest

Executa uma verificação de integridade que verifica se o firmware no qual o cluster é executado não foi adulterado.

Padrão: sem verificação de atestado.

Obrigatório: não

-nex

Torna a chave não extraível. A chave gerada não pode ser [exportada do HSM](#).

Padrão: a chave é extraível.

Obrigatório: não

-m

O valor que representa o mecanismo de encapsulamento. O CloudHSM é compatível com os seguintes mecanismos:

Mecanismo	Valor
AES_KEY_WRAP_PAD_PKCS5	4
NIST_AES_WRAP_NO_PAD	5
NIST_AES_WRAP_PAD	6
RSA_AES	7
RSA_OAEP (para obter o tamanho máximo dos dados, consulte a observação mais adiante nesta seção)	8

Mecanismo	Valor
AES_GCM	10
CLOUDHSM_AES_GCM	11
RSA_PKCS (para obter o tamanho máximo dos dados, consulte a observação mais adiante nesta seção). Consulte a nota 1 abaixo para ver uma mudança futura.	12

Obrigatório: Sim

 Note

Ao usar o mecanismo de RSA_OAEP agrupamento, o tamanho máximo da chave que você pode agrupar é determinado pelo módulo da chave RSA e pelo comprimento do hash especificado da seguinte forma: Tamanho máximo da chave = $\text{modulusLengthInBytes} - (\text{hashLengthIn2} * \text{Bytes}) - 2$.

Ao usar o mecanismo de encapsulamento RSA_PKCS, o tamanho máximo da chave que você pode encapsular é determinado pelo módulo da chave RSA da seguinte forma: Tamanho máximo da chave = $(\text{Bytes} - 11) * \text{modulusLengthIn}$

-t

Algoritmo hash	Valor
SHA1	2
SHA256	3
SHA384	4
SHA512	5
SHA224 (válido para RSA_AES e RSA_OAEP mecanismos)	6

Obrigatório: não

-noheader

Se estiver desencapsulando uma chave que foi encapsulada fora de `key_mgmt_util`, você deverá especificar esse parâmetro e todos os outros parâmetros associados.

Obrigatório: não

 Note

Se especificar esse parâmetro, você também deverá especificar os seguintes parâmetros `-noheader`:

- -l

Especifica o rótulo a ser adicionado à chave desencapsulada.

Obrigatório: Sim

- -kc

Especifica a classe da chave a ser desencapsulada. Os seguintes valores são aceitáveis:

3 = chave privada de um par de chaves pública/privada

4 = chave secreta (simétrica).

Obrigatório: Sim

- -kt

Especifica o tipo da chave a ser desencapsulada. Os seguintes valores são aceitáveis:

0 = RSA

1 = DSA

3 = ECC

16 = GENERIC_SECRET

21 = DES3

31 = AES

Obrigatório: Sim

Você também pode opcionalmente especificar os seguintes parâmetros `-noheader` :

- `-id`

O ID a ser adicionado à chave desencapsulada.

Obrigatório: não

- `-i`

O desencapsulamento do vetor de inicialização (IV) a ser usado.

Obrigatório: não

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Tópicos relacionados

- [wrapKey](#)
- [exSymKey](#)
- [imSymKey](#)

Verifique a assinatura de um arquivo usando o AWS CloudHSM KMU

Use o `verify` comando no AWS CloudHSM `key_mgmt_util` para confirmar se um arquivo foi assinado ou não por uma determinada chave. Para fazer isso, o comando `verify` compara um arquivo assinado com um arquivo de origem e analisa se eles estão criptograficamente relacionados com base em uma determinada chave pública e no mecanismo de assinatura. Os arquivos podem ser conectados AWS CloudHSM com a [sign](#) operação.

Os mecanismos de assinatura são representados pelos inteiros listados na seção de [parâmetros](#).

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
verify -h

verify -f <message-file>
       -s <signature-file>
       -k <public-key-handle>
       -m <signature-mechanism>
```

Exemplo

Esses exemplos mostram como usar `verify` para verificar se uma determinada chave pública foi usada para assinar um determinado arquivo.

Example: Verificar a assinatura de um arquivo

Esse comando tenta verificar se um arquivo chamado `hardwareCert.crt` foi assinado pela chave pública 262276 usando o mecanismo de assinatura `SHA256_RSA_PKCS` para produzir o arquivo assinado `hardwareCertSigned`. Como os parâmetros fornecidos representam um relacionamento de assinatura verdadeiro, o comando retorna uma mensagem de êxito.

```
Command: verify -f hardwareCert.crt -s hardwareCertSigned -k 262276 -m 1
```

```
Signature verification successful
```

```
Cfm3Verify returned: 0x00 : HSM Return: SUCCESS
```

Example: provar relacionamento de assinatura falso

Esse comando verifica se um arquivo chamado `hardwareCert.crt` foi assinado pela chave pública 262276 usando o mecanismo de assinatura `SHA256_RSA_PKCS` para produzir o arquivo assinado `userCertSigned`. Como os parâmetros fornecidos não formam um relacionamento de assinatura verdadeiro, o comando retorna uma mensagem de erro.

```
Command: verify -f hardwarecert.crt -s usercertsigned -k 262276 -m 1
```

```
Cfm3Verify returned: 0x1b
```

```
CSP Error: ERR_BAD_PKCS_DATA
```

Parâmetros

Esse comando usa os seguintes parâmetros.

-f

O nome do arquivo de mensagens de origem.

Obrigatório: Sim

-s

O nome do arquivo assinado.

Obrigatório: sim

-k

O identificador da chave pública que é considerado para ser usado para assinar o arquivo.

Obrigatório: Sim

-m

Um inteiro que representa o mecanismo de assinatura proposto usado para assinar o arquivo. Os mecanismos possíveis correspondem aos seguintes números inteiros:

Mecanismo de assinatura	Número inteiro correspondente
SHA1_RSA_PKCS	0
SHA256_RSA_PKCS	1
SHA384_RSA_PKCS	2
SHA512_RSA_PKCS	3
SHA224_RSA_PKCS	4
SHA1_RSA_PKCS_PSS	5
SHA256_RSA_PKCS_PSS	6
SHA384_RSA_PKCS_PSS	7

Mecanismo de assinatura	Número inteiro correspondente
SHA512_RSA_PKCS_PSS	8
SHA224_RSA_PKCS_PSS	9
ECDSA_SHA1	15
ECDSA_SHA224	16
ECDSA_SHA256	17
ECDSA_SHA384	18
ECDSA_SHA512	19

Obrigatório: Sim

Tópicos relacionados

- [sign](#)
- [getCert](#)
- [Gerar chaves](#)

Exportar uma AWS CloudHSM chave usando o KMU

Use o `wrapKey` comando no AWS CloudHSM `key_mgmt_util` para exportar uma cópia criptografada de uma chave simétrica ou privada do módulo de segurança de hardware (HSM) para um arquivo. Ao executar `wrapKey`, você especifica a chave a ser exportada, uma chave no HSM para criptografar (encapsular) a chave que deseja exportar e o arquivo de saída.

O comando `wrapKey` grava a chave criptografada em um arquivo que você especifica, mas não remove a chave do HSM ou impede que você a use em operações de criptografia. É possível exportar a mesma chave várias vezes.

Somente o proprietário de uma chave, ou seja, o usuário CU que a criou, pode exportá-la. Os usuários que compartilham a chave podem usá-la em operações de criptografia, mas não podem exportá-la.

Para importar a chave criptografada novamente no HSM, use [unWrapKey](#). Para exportar uma chave de texto simples de um HSM, use [exSymKey](#) ou [exportPrivateKey](#) conforme apropriado. O [aesWrapUnwrap](#) comando não pode descriptografar (desempacotar) as chaves que criptografam. `wrapKey`

Antes de executar um comando `key_mgmt_util`, você deve [iniciar key_mgmt_util](#) e [fazer login](#) no HSM como um usuário de criptografia (CU).

Sintaxe

```
wrapKey -h

wrapKey -k <exported-key-handle>
        -w <wrapping-key-handle>
        -out <output-file>
        [-m <wrapping-mechanism>]
        [-aad <additional authenticated data filename>]
        [-t <hash-type>]
        [-noheader]
        [-i <wrapping IV>]
        [-iv_file <IV file>]
        [-tag_size <num_tag_bytes>]
```

Exemplo

Example

Esse comando exporta uma chave simétrica Triple DES (3DES) de 192 bits (identificador de chave 7). Ele usa uma chave AES de 256 bits no HSM (identificador de chave 14) para encapsular a chave 7. Em seguida, ele grava a chave 3DES criptografada no arquivo `3DES-encrypted.key`.

A saída mostra que a chave 7 (3DES) foi encapsulada e gravada com sucesso no arquivo especificado. A chave criptografada tem 307 bytes de comprimento.

```
Command: wrapKey -k 7 -w 14 -out 3DES-encrypted.key -m 4
```

```
Key Wrapped.
```

```
Wrapped Key written to file "3DES-encrypted.key" length 307
```

```
Cfm2WrapKey returned: 0x00 : HSM Return: SUCCESS
```

Parâmetros

-h

Exibe a ajuda referente ao comando.

Obrigatório: Sim

-k

O identificador da chave que você deseja exportar. Digite o identificador da uma chave simétrica ou particular de sua propriedade. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para verificar se uma chave pode ser exportada, use o comando [getAttribute](#) para obter o valor do atributo OBJ_ATTR_EXTRACTABLE, que é representado pela constante 354. Para obter ajuda sobre a interpretação dos principais atributos, consulte [AWS CloudHSM referência de atributo-chave para KMU](#).

Você pode exportar apenas as chaves que você possui. Para encontrar o proprietário de uma chave, use o [getKeyInfo](#) comando.

Obrigatório: Sim

-w

Especifica a chave de empacotamento. Insira o identificador de uma chave AES ou chave RSA no HSM. Esse parâmetro é obrigatório. Para encontrar os identificadores de chave, use o comando [findKey](#).

Para criar uma chave de empacotamento, use [genSymKey](#) para gerar uma chave AES (tipo 31) ou [gen RSAKey Pair para gerar um par](#) de chaves RSA (tipo 0). Se você estiver usando um par de chaves RSA, certifique-se de encapsular a chave com uma das chaves e desencapsular com a outra. Para determinar se uma chave pode ser usada como uma chave de empacotamento, use [getAttribute](#) para obter o valor do atributo OBJ_ATTR_WRAP, que é representado pela constante 262.

Obrigatório: Sim

-out

O caminho e o nome do arquivo de saída. Quando o comando é bem-sucedido, esse arquivo contém uma cópia criptografada da chave exportada. Se o arquivo já existir, o comando o sobrescreverá sem aviso prévio.

Obrigatório: Sim

-m

O valor que representa o mecanismo de encapsulamento. O CloudHSM é compatível com os seguintes mecanismos:

Mecanismo	Valor
AES_KEY_WRAP_PAD_PKCS5	4
NIST_AES_WRAP_NO_PAD	5
NIST_AES_WRAP_PAD	6
RSA_AES	7
RSA_OAEP (para obter o tamanho máximo dos dados, consulte a observação mais adiante nesta seção)	8
AES_GCM	10
CLOUDHSM_AES_GCM	11
RSA_PKCS (para obter o tamanho máximo dos dados, consulte a observação mais adiante nesta seção). Consulte a nota 1 abaixo para ver uma mudança futura.	12

Obrigatório: Sim

Note

Ao usar o mecanismo de RSA_OAEP agrupamento, o tamanho máximo da chave que você pode agrupar é determinado pelo módulo da chave RSA e pelo comprimento do hash especificado da seguinte forma: Tamanho máximo da chave = $(\text{modulusLengthInBytes} - 2 * \text{Bytes} - 2) \cdot \text{hashLengthIn}$

Ao usar o mecanismo de encapsulamento RSA_PKCS, o tamanho máximo da chave que você pode encapsular é determinado pelo módulo da chave RSA da seguinte forma: Tamanho máximo da chave = $(\text{Bytes} - 11) \cdot \text{modulusLengthIn}$

-t

O valor que representa o algoritmo hash. O CloudHSM é compatível com os seguintes algoritmos:

Algoritmo hash	Valor
SHA1	2
SHA256	3
SHA384	4
SHA512	5
SHA224 (válido para RSA_AES e RSA_OAEP mecanismos)	6

Obrigatório: não

-aad

O nome do arquivo que contém AAD.

Note

Válido apenas para os mecanismos AES_GCM e CLOUDHSM_AES_GCM.

Obrigatório: não

-noheader

Omite o cabeçalho que especifica os [atributos de chave](#) específicos do CloudHSM. Use este parâmetro somente se desejar desencapsular a chave com ferramentas fora da `key_mgmt_util`.

Obrigatório: não

-i

O vetor de inicialização (IV) (valor hexadecimal).

Note

Válido somente quando passado com o parâmetro `-noheader` para mecanismos CLOUDHSM_AES_KEY_WRAP e NIST_AES_WRAP.

Obrigatório: não

-iv_file

O arquivo no qual você deseja gravar o valor IV obtido em resposta.

Note

Válido somente quando passado com o parâmetro `-noheader` para o mecanismo AES_GCM.

Obrigatório: não

-tag_size

O tamanho da tag a ser salva junto com o blob encapsulado.

Note

Válido somente quando passado com o parâmetro `-noheader` para mecanismos AES_GCM e CLOUDHSM_AES_GCM. O tamanho mínimo da tag é oito.

Obrigatório: não

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Tópicos relacionados

- [exSymKey](#)
- [imSymKey](#)
- [unWrapKey](#)

AWS CloudHSM referência de atributo-chave para KMU

Os comandos AWS CloudHSM `key_mgmt_util` usam constantes para representar os atributos das chaves em um módulo de segurança de hardware (HSM). Este tópico pode ajudá-lo a identificar os atributos, encontrar as constantes que os representam nos comandos e entender seus valores.

Você define os atributos de uma chave ao criá-la. Para alterar o atributo de token, que indica se uma chave é persistente ou existe apenas na sessão, use o comando [setAttribute](#) na `key_mgmt_util`. Para alterar os atributos de rótulos, encapsulamento, desencapsulamento, criptografia e descryptografia, use o comando `setAttribute` em `cloudhsm_mgmt_util`.

Para obter uma lista de atributos e suas constantes, use [listAttributes](#). Para obter os valores dos atributos de uma chave, use [getAttribute](#).

A tabela a seguir lista os atributos-chave, suas constantes e seus valores válidos.

Atributo	Constante	Valores
OBJ_ATTR_ALL	512	Representa todos os atributos.
OBJ_ATTR_ALWAYS_SENSITIVE	357	0: False. 1: True.
OBJ_ATTR_CLASS	0	2: Chave pública em um par de chaves privada pública. 3: Chave privada em um par de chaves privada pública.

Atributo	Constante	Valores
		4: Chave secreta (simétrica).
OBJ_ATTR_DECRYPT	261	0: False. 1: True. A chave pode ser usada para descriptografar dados.
OBJ_ATTR_DERIVE	268	0: False. 1: True. A função gera a chave.
OBJ_ATTR_DESTROYABLE	370	0: False. 1: True.
OBJ_ATTR_ENCRYPT	260	0: False. 1: True. A chave pode ser usada para criptografar dados.
OBJ_ATTR_EXTRACTABLE	354	0: False. 1: True. A chave pode ser exportada do HSMs.
OBJ_ATTR_ID	258	String definida pelo usuário. Deve ser exclusivo no cluster. O padrão é uma string vazia.
OBJ_ATTR_KCV	371	Valor de verificação da chave. Para obter mais informações, consulte Detalhes adicionais .

Atributo	Constante	Valores
OBJ_ATTR_KEY_TYPE	256	0: RSA. 1: DSA. 3: EC. 16: Segredo genérico. 18: RC4. 21: Triple DES (3DES). 31: AES.
OBJ_ATTR_LABEL	3	String definida pelo usuário. Ele não precisa ser exclusivo no cluster.
OBJ_ATTR_LOCAL	355	0. Falso. A chave foi importada para HSMs o. 1: True.
OBJ_ATTR_MODULUS	288	O módulo que foi usado para gerar um par de chaves RSA. Para chaves EC, esse valor representa a codificação DER do valor ANSI X9.62 “Q” em formato ECPoint hexadecimal. Para outros tipos de chave, esse atributo não existe.

Atributo	Constante	Valores
OBJ_ATTR_MODULUS_BITS	289	O comprimento do módulo utilizado para gerar um par de chaves RSA. Para chaves EC, isso representa o ID da curva elíptica usada para gerar a chave. Para outros tipos de chave, esse atributo não existe.
OBJ_ATTR_NEVER_EXPORTABLE	356	0: False. 1: True. A chave não pode ser exportada do HSMs.
OBJ_ATTR_PUBLIC_EXPONENT	290	O expoente público usado para gerar um par de chaves RSA. Para outros tipos de chave, esse atributo não existe.
OBJ_ATTR_PRIVATE	2	0: False. 1: True. Este atributo indica se os usuários não autenticados podem listar os atributos da chave. Como o provedor PKCS#11 do CloudHSM atualmente não oferece suporte a sessões públicas, todas as chaves (incluindo chaves públicas em um par de chaves públicas/privadas) têm esse atributo definido como 1.

Atributo	Constante	Valores
OBJ_ATTR_SENSITIVE	259	0: False. Chave pública em um par de chaves privada-pública. 1: True.
OBJ_ATTR_SIGN	264	0: False. 1: True. A chave pode ser usada para assinatura (chaves privadas).
OBJ_ATTR_TOKEN	1	0: False. Chave de sessão. 1: True. Chave persistente.
OBJ_ATTR_TRUSTED	134	0: False. 1: True.
OBJ_ATTR_UNWRAP	263	0: False. 1: True. A chave pode ser usada para descriptografar chaves.
OBJ_ATTR_UNWRAP_TEMPLATE	1073742354	Os valores devem usar o modelo de atributo aplicado a todas as chaves desencapsuladas com essa chave de encapsulamento.
OBJ_ATTR_VALUE_LEN	353	Tamanho da chave em bytes.

Atributo	Constante	Valores
OBJ_ATTR_VERIFY	266	0: False. 1: True. A chave pode ser usada para verificação (chaves públicas).
OBJ_ATTR_WRAP	262	0: False. 1: True. A chave pode ser usada para criptografar chaves.
OBJ_ATTR_WRAP_TEMPLATE	1073742353	Os valores devem usar o modelo de atributo para corresponder à chave agrupada usando essa chave de agrupamento.
OBJ_ATTR_WRAP_WITH_TRUSTED	528	0: False. 1: True.

Detalhes adicionais

Valor de verificação de chave (KCV)

O key check value (KCV – valor de verificação de chave) é um hash de 3 bytes ou soma de verificação de uma chave gerada quando o HSM importa ou gera uma chave. Você também pode calcular um KCV fora do HSM, como depois de exportar uma chave. Em seguida, é possível comparar os valores do KCV para confirmar a identidade e a integridade da chave. Para obter o KCV de uma chave, use [getAttribute](#).

AWS CloudHSM usa o seguinte método padrão para gerar um valor de verificação de chave:

- Chaves simétricas: primeiros 3 bytes do resultado da criptografia de um bloco zero com a chave.
- Pares de chaves assimétricas: primeiros 3 bytes do hash SHA-1 da chave pública.

- Chaves HMAC: o KCV para chaves HMAC não é suportado no momento.

Operações de descarga com o cliente AWS CloudHSM SDKs

Use um SDK de cliente para transferir operações criptográficas de aplicativos baseados em plataforma ou linguagem para módulos de segurança de hardware (HSMs).

AWS CloudHSM oferece duas versões principais, e o Client SDK 5 é a mais recente. Ele oferece uma variedade de vantagens em relação ao Client SDK 3 (a série anterior). Para obter mais informações, consulte [Benefícios do Client SDK 5](#). Para obter mais informações sobre o suporte de plataforma, consulte [AWS CloudHSM Plataformas compatíveis com o Client SDK 5](#).

Os tópicos a seguir descrevem como trabalhar com o AWS CloudHSM Client SDKs.

AWS CloudHSM suporta os seguintes componentes:

[the section called “Biblioteca PKCS #11”](#)

O PKCS #11 é um padrão para realizar operações criptográficas em módulos de segurança de hardware (HSMs). AWS CloudHSM oferece implementações da biblioteca PKCS #11 que são compatíveis com a versão 2.40 do PKCS #11.

[the section called “Mecanismo dinâmico do OpenSSL”](#)

O AWS CloudHSM OpenSSL Dynamic Engine permite que você descarregue operações criptográficas em seu cluster CloudHSM por meio da API OpenSSL.

[the section called “Provedor OpenSSL”](#)

O AWS CloudHSM OpenSSL Dynamic Engine Provider permite que você descarregue operações criptográficas em seu cluster CloudHSM por meio da API do OpenSSL Provider.

[the section called “Provedor JCE”](#)

O provedor AWS CloudHSM JCE é compatível com a Java Cryptographic Architecture (JCA). O provedor permite que você execute operações criptográficas no HSM.

[the section called “Provedor de armazenamento de chaves \(KSP\)”](#)

O AWS CloudHSM cliente para Windows inclui provedores de CNG e KSP.

Tópicos

- [Verifique sua versão AWS CloudHSM do SDK do cliente](#)
- [Compare AWS CloudHSM o suporte a componentes do SDK do cliente](#)
- [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#)
- [Usando o Client SDK 5 para trabalhar com AWS CloudHSM](#)
- [Usando a versão anterior do SDK para trabalhar com AWS CloudHSM](#)

Verifique sua versão AWS CloudHSM do SDK do cliente

Use os comandos a seguir para verificar a versão do Client SDK do que você está usando com o AWS CloudHSM.

Amazon Linux

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

Amazon Linux 2

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

CentOS 6

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

CentOS 7

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

CentOS 8

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

RHEL 6

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

RHEL 7

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

RHEL 8

Use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

Ubuntu 16.04 LTS

Use o seguinte comando:

```
apt list --installed | grep ^cloudhsm
```

Ubuntu 18.04 LTS

Use o seguinte comando:

```
apt list --installed | grep ^cloudhsm
```

Ubuntu 20.04 LTS

Use o seguinte comando:

```
apt list --installed | grep ^cloudhsm
```

Windows Server

Use o seguinte comando:

```
wmic product get name,version
```

Compare AWS CloudHSM o suporte a componentes do SDK do cliente

Além das ferramentas de linha de comando, o Client SDK 3 contém componentes que permitem o descarregamento de operações criptográficas para o HSM a partir de diversas plataformas ou aplicativos baseados em linguagem. O Client SDK 5 tem paridade com o Client SDK 3, exceto que ele ainda não oferece suporte aos provedores de CNG e KSP. A tabela a seguir compara a disponibilidade dos componentes no Client SDK 3 e no Client SDK 5.

Componente	Client SDK 5	Client SDK 3
Biblioteca PKCS #11	Sim	Sim
Provedor JCE	Sim	Sim
Mecanismo dinâmico do OpenSSL	Sim	Sim
Provedor de armazenamento de chaves (KSP)	Sim	Sim
CloudHSM Management Utility (CMU – utilitário de gerenciamento do CloudHSM) ¹	Sim	Sim
Key Management Utility (KMU – utilitário de gerenciamento de chaves) ¹	Sim	Sim
Configure a ferramenta	Sim	Sim

[1] Os componentes CMU e KMU estão incluídos na CLI do CloudHSM com o Client SDK 5.

As seções a seguir descrevem os componentes.

Biblioteca PKCS #11

O PKCS #11 é um padrão para realizar operações criptográficas em módulos de segurança de hardware (HSMs). AWS CloudHSM oferece implementações da biblioteca PKCS #11 que são compatíveis com a versão 2.40 do PKCS #11.

- Para o Client SDK 3, a biblioteca PKCS #11 é um componente exclusivo do Linux que corresponde ao suporte básico do Linux. Para obter mais informações, consulte [the section called “Suporte no Linux para Client SDK 3”](#).
- Para o Client SDK 5, a biblioteca PKCS #11 é um componente multiplataforma que corresponde ao suporte básico do Linux e do Windows Client SDK 5. Para obter mais informações, consulte [the section called “Suporte no Linux para o Client SDK 5 do ”](#) e [the section called “Suporte no Windows para o Client SDK 5 do ”](#).

CloudHSM Management Utility (CMU – utilitário de gerenciamento do CloudHSM)

A ferramenta de linha de comando do CloudHSM Management Utility (CMU) ajuda os agentes de criptografia a gerenciar usuários no HSM. Inclui ferramentas que criam, excluem e listam usuários e alteram senhas de usuários. Para obter mais informações, consulte [AWS CloudHSM Utilitário de gerenciamento \(CMU\)](#).

Key Management Utility (KMU – utilitário de gerenciamento de chaves)

O Key Management Utility (KMU) é uma ferramenta de linha de comando que ajuda os usuários de criptografia a gerenciar chaves nos HSMs. Para obter mais informações, consulte [AWS CloudHSM Utilitário de gerenciamento de chaves \(KMU\)](#).

Provedor JCE

O provedor AWS CloudHSM JCE é compatível com a Java Cryptographic Architecture (JCA). O provedor permite que você execute operações criptográficas no HSM.

O provedor JCE é um componente exclusivo do Linux que corresponde ao suporte básico do Linux. Para obter mais informações, consulte [the section called “Suporte no Linux para Client SDK 3”](#).

- Para o Client SDK 3, é necessário o OpenJDK 1.8

Mecanismo dinâmico do OpenSSL

O AWS CloudHSM OpenSSL Dynamic Engine permite que você descarregue operações criptográficas em seu cluster CloudHSM por meio da API OpenSSL.

- Para o Client SDK 3, o OpenSSL Dynamic Engine é o único componente do Linux que não corresponde ao suporte básico do Linux. Veja as exclusões abaixo.
 - Requer OpenSSL 1.0.2[f+]

Plataformas não suportadas:

- CentOS 8
- Red Hat Enterprise Linux (RHEL) 8
- Ubuntu 18.04 LTS

Essas plataformas são fornecidas com uma versão do OpenSSL incompatível com o OpenSSL Dynamic Engine para Client SDK 3. O AWS CloudHSM suporta essas plataformas com o OpenSSL Dynamic Engine para Client SDK 5.

- Para o Client SDK 5, o OpenSSL Dynamic Engine é um componente exclusivo do Linux que requer OpenSSL 1.0.2, 1.1.1 ou 3.x.

Provedor de armazenamento de chaves (KSP)

O Provedor de armazenamento de chaves (KSP) é uma API de criptografia específica para o sistema operacional Microsoft Windows.

Para o Client SDK 3, os provedores de CNG e KSP são um componente exclusivo do Windows que corresponde ao suporte básico do Windows. Para obter mais informações, consulte [Suporte do Windows para o AWS CloudHSM Client SDK 3](#).

Para o Client SDK 5, o Provedor de armazenamento de chaves (KSP) é um s componente exclusivo do Windows que corresponde ao suporte básico do Windows. Para obter mais informações, consulte [Suporte do Windows para o AWS CloudHSM Client SDK 5](#).

Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5

Em AWS CloudHSM, os aplicativos do cliente realizam operações criptográficas usando o Kit de Desenvolvimento de Software AWS CloudHSM do Cliente (SDK). O Client SDK 5 é o SDK principal que sempre recebe novos atributos e suporte de plataforma.

O Client SDK 3 inclui duas ferramentas de linha de comando separadas: o CMU para gerenciar usuários e a KMU para gerenciar chaves e realizar operações com chaves. O Client SDK 5 consolida as funções do CMU e do KMU (ferramentas que foram oferecidas com o Client SDK 3) em uma única ferramenta, a [AWS CloudHSM Interface de linha de comando \(CLI\)](#). As operações de gerenciamento de usuários podem ser encontradas nos subcomandos contidos em [A categoria user na CLI do CloudHSM](#) e [A categoria de quórum na CloudHSM CLI](#). As operações de gerenciamento de chaves podem ser encontradas no [subcomando key](#) e as operações criptográficas podem ser encontradas no [subcomando crypto](#). Consulte [Referência para comandos da CLI do CloudHSM](#) para obter uma lista completa dos comandos.

Para conhecer os benefícios da migração, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

Consulte os tópicos a seguir para obter instruções detalhadas sobre a migração do Client SDK 3 para o Client SDK 5. A versão mais recente do SDK AWS CloudHSM do cliente é a 5.16.

- [Migre sua biblioteca AWS CloudHSM PKCS #11 do SDK do cliente 3 para o SDK do cliente 5](#)
- [Migre seu OpenSSL Dynamic Engine do AWS CloudHSM Client SDK 3 para o Client SDK 5](#)
- [Migre seu provedor de armazenamento de chaves \(KSP\) do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#)
- [Migre seu provedor de JCE do AWS CloudHSM Client SDK 3 para o Client SDK 5](#)

Para funcionalidades ou casos de uso que não são compatíveis com a CloudHSM CLI, entre em contato com [AWS Support](#).

Migre sua biblioteca AWS CloudHSM PKCS #11 do SDK do cliente 3 para o SDK do cliente 5

Use este tópico para migrar sua [biblioteca AWS CloudHSM PKCS #11](#) do SDK do cliente 3 para o SDK do cliente 5. Para conhecer os benefícios da migração, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

Em AWS CloudHSM, os aplicativos do cliente realizam operações criptográficas usando o Kit de Desenvolvimento de Software AWS CloudHSM do Cliente (SDK). O Client SDK 5 é o SDK principal que sempre recebe novos atributos e suporte de plataforma.

Para revisar as instruções de migração para todos os provedores, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Preparar-se abordando as mudanças mais importantes

Revise essas alterações importantes e atualize a aplicação em seu ambiente de desenvolvimento adequadamente.

Os mecanismos de encapsulamento foram alterados

Mecanismo do Client SDK 3	Mecanismo equivalente do Client SDK 5
CKM_AES_KEY_WRAP	CKM_CLOUDHSM_AES_KEY_WRAP_P KCS5_PAD
CKM_AES_KEY_WRAP_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_Z ERO_PAD
CKM_CLOUDHSM_AES_KEY_WRAP_P KCS5_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_P KCS5_PAD
CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD
CKM_CLOUDHSM_AES_KEY_WRAP_Z ERO_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_Z ERO_PAD

ECDH

No Client SDK 3, você pode usar o ECDH e especificar um KDF. No momento, essa funcionalidade não está disponível no Client SDK 5. Se sua aplicação precisar dessa funcionalidade, entre em contato com o [suporte](#).

Os manipuladores de chaves agora são específicos da sessão

Para usar com sucesso os identificadores de chave no Client SDK 5, você deve obter identificadores de chave sempre que executar um aplicativo. Se você tiver aplicações existentes que usarão os

mesmos identificadores de chave em diferentes execuções, você deve modificar seu código para obter o identificador de chave sempre que executar a aplicação. Para obter informações sobre como recuperar identificadores de chave, consulte [este exemplo de AWS CloudHSM PKCS #11](#). Essa alteração está em conformidade com a especificação [PKCS #11 2.40](#).

Migrar para o Client SDK 5

Siga as instruções nesta seção para migrar do Client SDK 3 para o Client SDK 5.

Note

No momento, o Amazon Linux, o Ubuntu 16.04, o Ubuntu 18.04, o CentOS 6, o CentOS 8 e o RHEL 6 não são compatíveis com o Client SDK 5. Se você estiver usando uma dessas plataformas com o Client SDK 3, precisará escolher uma plataforma diferente ao migrar para o Client SDK 5.

1. Desinstale a biblioteca PKCS #11 para Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client-pkcs11
```

CentOS 7

```
$ sudo yum remove cloudhsm-client-pkcs11
```

RHEL 7

```
$ sudo yum remove cloudhsm-client-pkcs11
```

RHEL 8

```
$ sudo yum remove cloudhsm-client-pkcs11
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client-pkcs11
```


Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client-pkcs11
```

2. Pare o Client Daemon para o Client SDK 3.

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

3. Desinstale o Client Daemon do Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client
```

CentOS 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 8

```
$ sudo yum remove cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Note

As configurações personalizadas precisam ser habilitadas novamente.

4. Instale a biblioteca PKCS #11 do Client SDK seguindo as etapas em [Instale a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#).
5. O Client SDK 5 apresenta um novo formato de arquivo de configuração e uma ferramenta de inicialização de linha de comando. Para inicializar sua biblioteca PKCS #11 do Client SDK 5, siga as instruções listadas no guia do usuário em [Bootstrap o Client SDK](#).
6. Teste sua aplicação em seu ambiente de desenvolvimento. Faça atualizações no código existente para resolver as alterações importantes antes da migração final.

Tópicos relacionados

- [Melhores práticas para AWS CloudHSM](#)

Migre seu OpenSSL Dynamic Engine do AWS CloudHSM Client SDK 3 para o Client SDK 5

Use este tópico para migrar seu [OpenSSL Dynamic Engine](#) do Client SDK 3 do AWS CloudHSM para o Client SDK 5. Para conhecer os benefícios da migração, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

Em AWS CloudHSM, os aplicativos do cliente realizam operações criptográficas usando o Kit de Desenvolvimento de Software AWS CloudHSM do Cliente (SDK). O Client SDK 5 é o SDK principal que sempre recebe novos atributos e suporte de plataforma.

Note

Atualmente, não há suporte para a geração de números aleatórios no Client SDK 5 com OpenSSL Dynamic Engine.

Para revisar as instruções de migração para todos os provedores, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Migrar para o Client SDK 5

Siga as instruções nesta seção para migrar do Client SDK 3 para o Client SDK 5.

Note

No momento, o Amazon Linux, o Ubuntu 16.04, o Ubuntu 18.04, o CentOS 6, o CentOS 8 e o RHEL 6 não são compatíveis com o Client SDK 5. Se você estiver usando uma dessas plataformas com o Client SDK 3, precisará escolher uma plataforma diferente ao migrar para o Client SDK 5.

1. Desinstale o OpenSSL Dynamic Engine para Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client-dyn
```

CentOS 7

```
$ sudo yum remove cloudhsm-client-dyn
```

RHEL 7

```
$ sudo yum remove cloudhsm-client-dyn
```

RHEL 8

```
$ sudo yum remove cloudhsm-client-dyn
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client-dyn
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client-dyn
```

2. Pare o Client Daemon para o Client SDK 3.

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

3. Desinstale o Client Daemon do Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client
```

CentOS 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 8

```
$ sudo yum remove cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client
```

 Note

As configurações personalizadas precisam ser habilitadas novamente.

4. Instale o OpenSSL Dynamic Engine do Client SDK seguindo as etapas em [Instale o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5](#).
5. O Client SDK 5 apresenta um novo formato de arquivo de configuração e uma ferramenta de inicialização de linha de comando. Para inicializar seu OpenSSL Dynamic Engine do Client SDK 5, siga as instruções listadas no guia do usuário em [Bootstrap o Client SDK](#).
6. Teste sua aplicação em seu ambiente de desenvolvimento. Faça atualizações no código existente para resolver as alterações importantes antes da migração final.

Tópicos relacionados

- [Melhores práticas para AWS CloudHSM](#)

Migre seu provedor de armazenamento de chaves (KSP) do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5

Este tópico explica como migrar seu [Provedor de armazenamento de chaves \(KSP\)](#) do Client SDK 3 do AWS CloudHSM para o Client SDK 5. A versão mais recente do SDK AWS CloudHSM do cliente é a 5.16. Para obter informações sobre os benefícios da migração, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

Em AWS CloudHSM, você usa o AWS CloudHSM Client Software Development Kit (SDK) para realizar operações criptográficas. O Client SDK 5 é o SDK principal que recebe novos atributos e atualizações de suporte de plataforma.

Para obter as instruções de migração para todos os provedores, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Migrar para o Client SDK 5

1. Pare o Client Daemon para o Client SDK 3.

```
PS C:\> Stop-Service "AWS CloudHSM Client"
```

2. Instale o Provedor de armazenamento de chaves (KSP) do Client SDK 5 na sua instância do Windows Server. Para instruções, consulte [Instale o provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#).
3. Configure o Provedor de armazenamento de chaves (KSP) do Client SDK 5 usando o novo formato de arquivo de configuração e a ferramenta de inicialização de linha de comando. Para instruções, consulte [Bootstrap o Client SDK](#).
4. O Key Storage Provider (KSP) para o AWS CloudHSM Client SDK 5 inclui o modo de SDK3 compatibilidade para oferecer suporte aos principais arquivos de referência gerados em SDK3. Para obter mais informações, consulte [SDK3 modo de compatibilidade do Key Storage Provider \(KSP\) para AWS CloudHSM](#).

 Note

Você deve ativar o modo de SDK3 compatibilidade ao usar os arquivos de referência de chave gerados pelo Client SDK 3 com o Client SDK 5.

Migrar para novas instâncias do Windows Server

1. Conclua todas as etapas em [Migrar para o Client SDK 5](#) em suas novas instâncias do Windows Server.
2. Verificar arquivos de referência de chaves existentes

Em sua instância original do Windows Server, verifique os arquivos de referência chave em C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition.

- Se existirem arquivos de referência chave, copie todo o conteúdo em C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP incluindo GlobalPartition, para o mesmo caminho de diretório na sua nova instância do Windows Server. Crie o diretório, caso ele ainda não exista.
- Se os arquivos de referência chave não existirem, use `cloudhsm-cli key generate-file --encoding ksp-key-reference` em sua nova instância do Windows Server para criá-los. Para instruções, consulte [Gerar referências de chave KSP \(Windows\)](#).

3. Verificar um certificado raiz

Verifique seu certificado raiz nas autoridades de certificação raiz confiáveis:

```
PS C:\Users\Administrator\Desktop> certutil -store Root

Root "Trusted Root Certification Authorities"
===== Certificate 0 =====
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
NotBefore: 2/5/2020 1:38 PM
NotAfter: 2/5/2021 1:48 PM
Issuer: CN=MYRootCA
Signature matches Public Key
Root Certificate: Subject matches Issuer
Cert Hash(sha1): cert-hash
No key provider information
Cannot find the certificate and private key for decryption.
CertUtil: -store command completed successfully.
```

Note

Anote o número de série do certificado para uso na próxima etapa.

4. Exportar um certificado raiz

Exporte o certificado raiz para um arquivo:

```
certutil -store Root certificate-serial-number root-certificate-name.cer
```

5. Verificar o certificado de backend do HSM

Verifique seu certificado de backend do HSM no repositório de certificados pessoais:

```
PS C:\Users\Administrator\Desktop> certutil -store My

my "Personal"
===== Certificate 0 =====
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
NotBefore: 2/5/2020 1:38 PM
NotAfter: 2/5/2021 1:48 PM
```



```
Subject: CN=www.mydomain.com, OU=Certificate Management, O=Information Technology,
L=Houston, S=Texas, C=US
Non-root Certificate
Cert Hash(sha1): cert-hash
  Key Container = key-container-name
  Provider = Cavium Key Storage Provider
Private key is NOT exportable
Encryption test passed
CertUtil: -store command completed successfully.
```

Note

Anote o número de série do certificado para uso na próxima etapa.

6. Exportar certificado de backend do HSM

Exporte o certificado de backend do HSM para um arquivo:

```
certutil -store My certificate-serial-number signed-certificate-name.cer
```

7. Importar um certificado raiz

Na sua nova instância do Windows:

1. Copie o arquivo CA raiz para sua nova instância do Windows
2. Importe o certificado:

```
certutil -addstore Root root-certificate-name.cer
```

8. Verificar a instalação do certificado raiz

Confirme se o certificado raiz foi instalado corretamente:

```
PS C:\Users\Administrator\Desktop> certutil -store Root

Root "Trusted Root Certification Authorities"
===== Certificate 0 =====
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
NotBefore: 2/5/2020 1:38 PM
NotAfter: 2/5/2021 1:48 PM
Issuer: CN=MYRootCA
```

```
Signature matches Public Key
Root Certificate: Subject matches Issuer
Cert Hash(sha1): cert-hash
No key provider information
Cannot find the certificate and private key for decryption.
CertUtil: -store command completed successfully.
```

9. Importar um certificado de backend do HSM

Na sua nova instância do Windows:

1. Copie o certificado de backend do HSM para sua nova instância do Windows
2. Importe o certificado:

```
certutil -addstore My signed-certificate-name.cer
```

10. Verificar a instalação do certificado de backend do HSM

Confirme se o certificado de backend do HSM foi instalado corretamente:

```
PS C:\Users\Administrator\Desktop> certutil -store My

my "Personal"
===== Certificate 0 =====
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
  NotBefore: 2/5/2020 1:38 PM
  NotAfter: 2/5/2021 1:48 PM
Subject: CN=www.mydomain.com, OU=Certificate Management, O=Information Technology,
  L=Houston, S=Texas, C=US
Non-root Certificate
Cert Hash(sha1): cert-hash
No key provider information
Cannot find the certificate and private key for decryption.
CertUtil: -store command completed successfully.
```

Note

Anote o número de série do certificado para uso nas próximas etapas.

11. Criar um arquivo de referência de chaves (opcional)

Conclua esta etapa somente se precisar criar um novo arquivo de referência de chave. Caso contrário, siga para a próxima etapa.

 Note

Esse atributo só está disponível no SDK versão 5.16.0 e posterior.

1. Instale o [OpenSSL](#) e extraia o módulo:

```
openssl x509 -in signed-certificate-name.cer -modulus -noout
```

 Note

O comando OpenSSL gera o módulo no formato: `Modulus=modulus-value`.
Observe o *modulus-value* para uso no próximo comando.

2. Crie um arquivo de referência chave com a CloudHSM CLI, consulte: [Gerar referências de chave KSP \(Windows\)](#)

```
& "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" key generate-  
file --encoding ksp-key-reference --filter attr.class=private-key  
attr.modulus=0xmodulus-value
```

 Note

Os argumentos do comando *modulus-value* na CLI do CloudHSM devem ser 0x prefixados com para indicar o formato hexadecimal.
Os principais arquivos de referência são criados em `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`.

12. Criar configuração de restauração

Crie um arquivo chamado `repair.txt` com o seguinte conteúdo:

```
[Properties]  
11 = "" ; Add friendly name property  
2 = "{text}" ; Add Key Provider Information property
```

```
_continue_="Container=key-container-name&"
_continue_="Provider=Cavium Key Storage Provider&"
_continue_="Flags=0&"
_continue_="KeySpec=2"
```

 Note

key-container-name Substitua pelo nome do arquivo de referência chave de.
C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP
\GlobalPartition

13. Reparar o armazenamento de certificados

Execute o comando repair:

```
certutil -repairstore My certificate-serial-number repair.txt
```

 Note

O número de série do certificado é obtido nas etapas anteriores ao verificar a instalação do certificado de backend do HSM.

14. Verificar uma associação de certificados

Confirme se o certificado foi associado corretamente:

```
PS C:\Users\Administrator\Desktop> certutil -store My

my "Personal"
===== Certificate 0 =====
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
NotBefore: 2/5/2020 1:38 PM
NotAfter: 2/5/2021 1:48 PM
Subject: CN=www.mydomain.com, OU=Certificate Management, O=Information Technology,
L=Houston, S=Texas, C=US
Non-root Certificate
Cert Hash(sha1): cert-hash
Key Container = key-container-name
Provider = Cavium Key Storage Provider
```

```
Private key is NOT exportable  
ERROR: Could not verify certificate public key against private key  
CertUtil: -store command completed successfully.
```

Verifique se a saída mostra:

- O nome correto do contêiner de chaves
- O Cavium Key Storage Provider
- Esse ERROR: Could not verify certificate public key against private key é um problema conhecido, consulte [Problema: falha na verificação de um armazenamento de certificados](#)

15. Testar a aplicação

Antes de concluir a migração:

1. Teste sua aplicação em seu ambiente de desenvolvimento.
2. Atualize seu código para resolver quaisquer alterações importantes
3. Para obter orientação específica da aplicação, consulte [Integrando aplicativos de terceiros com AWS CloudHSM](#)

Verificar a migração

Depois de concluir as etapas de migração, verifique se:

- Seus certificados estão instalados corretamente nos repositórios de certificados corretos
- Os principais arquivos de referência estão presentes no local correto
- Sua aplicação pode realizar operações criptográficas usando os certificados migrados

Solução de problemas

Se você encontrar problemas durante a migração, verifique:

- Todos os certificados são exportados corretamente do sistema de origem
- Os números de série do certificado coincidem entre os sistemas
- Os nomes dos contêineres principais no arquivo repair.txt correspondem aos seus arquivos de referência de chave

- SDK3 o modo de compatibilidade é ativado se estiver usando arquivos SDK3 de referência de chave gerados

Tópicos relacionados

- [Melhores práticas para AWS CloudHSM](#)

Migre seu provedor de JCE do AWS CloudHSM Client SDK 3 para o Client SDK 5

Use este tópico para migrar seu [provedor JCE](#) do Client SDK 3 do AWS CloudHSM para o Client SDK 5. Para conhecer os benefícios da migração, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

Em AWS CloudHSM, os aplicativos do cliente realizam operações criptográficas usando o Kit de Desenvolvimento de Software AWS CloudHSM do Cliente (SDK). O Client SDK 5 é o SDK principal que sempre recebe novos atributos e suporte de plataforma.

O provedor Client SDK 3 JCE usa classes personalizadas e APIs que não fazem parte da especificação JCE padrão. O Client SDK 5 para o provedor JCE está em conformidade com a especificação JCE e é incompatível com versões anteriores do Client SDK 3 em determinadas áreas. As aplicações do cliente podem exigir alterações como parte da migração para o Client SDK 5. Esta seção descreve as alterações necessárias para uma migração bem-sucedida.

Para revisar as instruções de migração para todos os provedores, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Tópicos

- [Preparar-se abordando as mudanças mais importantes](#)
- [Migrar para o Client SDK 5](#)
- [Tópicos relacionados](#)

Preparar-se abordando as mudanças mais importantes

Revise essas alterações importantes e atualize a aplicação em seu ambiente de desenvolvimento adequadamente.

A classe e o nome do provedor foram alterados

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Classe e nome do provedor	A classe de provedor JCE no Client SDK 3 é chamada de <code>CaviumProvider</code> e tem o nome de provedor Cavium.	No Client SDK 5, a classe <code>Provider</code> é chamada de <code>CloudHsmP</code> rovider e tem o nome de provedor CloudHSM.	Um exemplo de como inicializar o <code>CloudHsmP</code> rovider objeto está disponível no repositório de AWS CloudHSM GitHub amostra .

O login explícito foi alterado, o implícito não

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Login explícito	O Client SDK 3 usa a classe <code>LoginManager</code> para o login explícito ¹ .	No Client SDK 5, o provedor CloudHSM implementa a <code>AuthProvider</code> para o login explícito. <code>AuthProvider</code> é uma classe Java padrão e segue a forma idiomática do Java de fazer login em um provedor. Com o gerenciamento aprimorado do estado de login no Client SDK 5, as aplicações não precisam mais monitorar e realizar	Para ver um exemplo de como usar o login explícito com o SDK 5 do cliente, consulte o <code>LoginRunner</code> exemplo no repositório de amostras do AWS GitHub CloudHSM .

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
		o login durante as reconexões ² .	
Login implícito	Nenhuma alteração é necessária para o login implícito. O mesmo arquivo de propriedades e todas as variáveis de ambiente continuarão funcionando para o login implícito ao migrar do Client SDK 3 para o Client SDK 5.		Para ver um exemplo de como usar o login implícito com o Client SDK 5, consulte a LoginRunner AWS CloudHSM GitHub amostra no repositório de amostras.

- [1] Trecho de código do Client SDK 3:

```

LoginManager lm = LoginManager.getInstance();

lm.login(partition, user, pass);

```

- [2] Trecho de código do Client SDK 5:

```

// Construct or get the existing provider object
AuthProvider provider = new CloudHsmProvider();

// Call login method on the CloudHsmProvider object
// Here loginHandler is a CallbackHandler
provider.login(null, loginHandler);

```

Para ver um exemplo de como usar o login explícito com o Client SDK 5, consulte a [LoginRunner AWS CloudHSM GitHub amostra no repositório](#) de amostras.

A geração de chaves mudou

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Geração de chaves	No Client SDK 3, <code>Cavium[Key-type]AlgorithmParameterSpec</code> é usado para especificar parâmetros de geração de chaves. Para obter um trecho de código, consulte a nota de rodapé 1 .	No Client SDK 5, <code>KeyAttributesMap</code> é usado para especificar atributos de geração de chaves. Para obter um trecho de código, consulte a nota de rodapé 2 .	Para ver um exemplo de como usar <code>KeyAttributesMap</code> para gerar uma chave simétrica, consulte a SymmetricKeys amostra no repositório de amostras do GitHub AWS CloudHSM.
Geração de pares de chaves	No Client SDK 3, <code>Cavium[Key-type]AlgorithmparameterSpec</code> é usado para especificar parâmetros de geração de pares de chaves. Para obter um trecho de código, consulte a nota de rodapé 3 .	No Client SDK 5, <code>KeyPairAttributesMap</code> é usado para especificar esses parâmetros. Para obter um trecho de código, consulte a nota de rodapé 4 .	Para ver um exemplo de como usar <code>KeyAttributesMap</code> para gerar uma chave assimétrica, consulte a AsymmetricKeys amostra no repositório de amostras de AWS CloudHSM GitHub amostras.

- [1] Trecho de código de geração de chaves do Client SDK 3:

```
KeyGenerator keyGen = KeyGenerator.getInstance("AES", "Cavium");
CaviumAESKeyGenParameterSpec aesSpec = new CaviumAESKeyGenParameterSpec(
    keySizeInBits,
    keyLabel,
    isExtractable,
```

```
isPersistent);
keyGen.init(aesSpec);
SecretKey aesKey = keyGen.generateKey();
```

- [2] Trecho de código de geração de chaves do Client SDK 5:

```
KeyGenerator keyGen = KeyGenerator.getInstance("AES",
CloudHsmProvider.PROVIDER_NAME);

final KeyAttributesMap aesSpec = new KeyAttributesMap();
aesSpec.put(KeyAttribute.LABEL, keyLabel);
aesSpec.put(KeyAttribute.SIZE, keySizeInBits);
aesSpec.put(KeyAttribute.EXTRACTABLE, isExtractable);
aesSpec.put(KeyAttribute.TOKEN, isPersistent);

keyGen.init(aesSpec);
SecretKey aesKey = keyGen.generateKey();
```

- [3] Trecho de código de geração de pares de chaves do Client SDK 3:

```
KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("rsa", "Cavium");
CaviumRSAKeyGenParameterSpec spec = new CaviumRSAKeyGenParameterSpec(
keySizeInBits,
new BigInteger("65537"),
label + ":public",
label + ":private",
isExtractable,
isPersistent);

keyPairGen.initialize(spec);

keyPairGen.generateKeyPair();
```

- [4] Trecho do código de geração de pares de chaves do Client SDK 5:

```
KeyPairGenerator keyPairGen =
KeyPairGenerator.getInstance("RSA", providerName);

// Set attributes for RSA public key
final KeyAttributesMap publicKeyAttrsMap = new KeyAttributesMap();
publicKeyAttrsMap.putAll(additionalPublicKeyAttributes);
publicKeyAttrsMap.put(KeyAttribute.LABEL, label + ":Public");
publicKeyAttrsMap.put(KeyAttribute.MODULUS_BITS, keySizeInBits);
```

```

publicKeyAttrsMap.put(KeyAttribute.PUBLIC_EXPONENT,
new BigInteger("65537").toByteArray());

// Set attributes for RSA private key
final KeyAttributesMap privateKeyAttrsMap = new KeyAttributesMap();
privateKeyAttrsMap.putAll(additionalPrivateKeyAttributes);
privateKeyAttrsMap.put(KeyAttribute.LABEL, label + ":Private");

// Create KeyPairAttributesMap and use that to initialize the
// keyPair generator
KeyPairAttributesMap keyPairSpec =
new KeyPairAttributesMapBuilder()
.withPublic(publicKeyAttrsMap)
.withPrivate(privateKeyAttrsMap)
.build();

keyPairGen.initialize(keyPairSpec);
keyPairGen.generateKeyPair();

```

As chaves de localização, exclusão e referência foram alteradas

Encontrar uma chave já gerada AWS CloudHSM envolve o uso do KeyStore. O SDK 3 do cliente tem dois KeyStore tipos: Cavium e CloudHSM. O SDK 5 do cliente tem apenas um KeyStore tipo: CloudHSM.

Cavium KeyStore Mudar do para CloudHSM KeyStore requer uma mudança de KeyStore tipo. Além disso, o Client SDK 3 usa identificadores de chave para fazer referência a chaves, enquanto o Client SDK 5 usa rótulos de chave. As mudanças de comportamento resultantes são as mencionadas a seguir.

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Referências de chave	Com o Client SDK 3, as aplicações usam rótulos ou identificadores de chaves para fazer referência a a chaves no HSM. Eles usam rótulos	No Client SDK 5, as aplicações podem usar o AWS CloudHSM KeyStore Classe Java para Client SDK 5 para encontrar chaves	

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
	com KeyStore para encontrar uma chave ou usam alças para criar CaviumKey objetos.	por rótulo. Para encontrar as chaves pela alça, use o AWS CloudHSM KeyStoreWithAttributes com AWS CloudHSM KeyReferenceSpec .	
Descobrir várias entradas	Ao pesquisar uma chave usando <code>getEntry</code> , <code>getKey</code> ou <code>getCertificate</code> em cenários em que existem vários itens com os mesmos critérios no Cavium KeyStore, somente a primeira entrada encontrada será retornada.	Com o AWS CloudHSM KeyStore e KeyStoreWithAttributes , esse mesmo cenário resultará no lançamento de uma exceção. Para corrigir esse problema, é recomendável definir rótulos exclusivos para chaves usando o comando Definir os atributos das chaves com a CloudHSM CLI na CLI do CloudHSM. Ou use KeyStoreWithAttributes#getKeys para retornar todas as chaves correspondentes aos critérios.	

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Encontrar todas as chaves	No Client SDK 3, é possível encontrar todas as chaves no HSM usando <code>Util.findAllKeys()</code> .	O Client SDK 5 torna a descoberta de chaves mais simples e eficiente usando a classe <code>KeyStoreWithAttributes</code> . Quando possível, armazene suas chaves em cache para minimizar a latência. Para obter mais informações, consulte Gerencie com eficácia as chaves em seu aplicativo . Quando precisar recuperar todas as chaves do HSM, use <code>KeyStoreWithAttributes#getKeys</code> com um <code>KeyAttributesMap</code> vazio.	Um exemplo que usa a <code>KeyStoreWithAttributes</code> classe para encontrar uma chave está disponível no repositório de AWS CloudHSM GitHub amostra e um trecho de código é exibido em. 1

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Exclusão de chaves	O Client SDK 3 usa <code>Util.deleteKey()</code> para excluir uma chave.	O objeto <code>Key</code> no Client SDK 5 implementa a interface <code>Destroyable</code> , que permite excluir as chaves usando o método <code>destroy()</code> dessa interface.	Um exemplo de código mostrando a funcionalidade de exclusão da chave pode ser encontrado no repositório de amostra do GitHub CloudHSM . Um trecho de amostra para cada SDK é mostrado em 2 .

- [1] Um trecho é mostrado abaixo:

```
KeyAttributesMap findSpec = new KeyAttributesMap();
findSpec.put(KeyAttribute.LABEL, label);
findSpec.put(KeyAttribute.KEY_TYPE, keyType);
KeyStoreWithAttributes keyStore = KeyStoreWithAttributes.getInstance("CloudHSM");

keyStore.load(null, null);
keyStore.getKey(findSpec);
```

- [2] Exclusão de uma chave no Client SDK 3:

```
Util.deleteKey(key);
```

Exclusão de uma chave no Client SDK 5:

```
((Destroyable) key).destroy();
```

As operações de desencapsulamento de cifras foram alteradas, outras operações de cifragem não

 Note

Nenhuma alteração é necessária para as operações de criptografia. encrypt/decrypt/wrap

As operações de desencapsulamento exigem que a classe `CaviumUnwrapParameterSpec` do Client SDK 3 seja substituída por uma das seguintes classes específicas das operações criptográficas listadas.

- `GCMUnwrapKeySpec` para desencapsulamento AES/GCM/NoPadding
- `IvUnwrapKeySpec` para AESWrap unwrap e AES/CBC/NoPadding unwrap
- `OAEPUnwrapKeySpec` para RSA OAEP unwrap

Trecho de exemplo para `OAEPUnwrapKeySpec`:

```
OAEPParameterSpec oaepParameterSpec =
new OAEPParameterSpec(
    "SHA-256",
    "MGF1",
    MGF1ParameterSpec.SHA256,
    PSpecified.DEFAULT);

KeyAttributesMap keyAttributesMap =
    new KeyAttributesMap(KeyAttributePermissiveProfile.KEY_CREATION);
keyAttributesMap.put(KeyAttribute.TOKEN, true);
keyAttributesMap.put(KeyAttribute.EXTRACTABLE, false);

OAEPUnwrapKeySpec spec = new OAEPUnwrapKeySpec(oaepParameterSpec,
    keyAttributesMap);

Cipher hsmCipher =
    Cipher.getInstance(
        "RSA/ECB/OAEPPadding",
        CloudHsmProvider.PROVIDER_NAME);
hsmCipher.init(Cipher.UNWRAP_MODE, key, spec);
```

As operações de assinatura não foram alteradas

Nenhuma alteração é necessária para as operações de assinatura.

Migrar para o Client SDK 5

Siga as instruções nesta seção para migrar do Client SDK 3 para o Client SDK 5.

Note

No momento, o Amazon Linux, o Ubuntu 16.04, o Ubuntu 18.04 CentOS 6, o CentOS 8 e o RHEL 6 não são compatíveis com o Client SDK 5. Se você estiver usando uma dessas plataformas com o Client SDK 3, precisará escolher uma plataforma diferente ao migrar para o Client SDK 5.

1. Desinstalar o provedor JCE para o Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client-jce
```

CentOS 7

```
$ sudo yum remove cloudhsm-client-jce
```

RHEL 7

```
$ sudo yum remove cloudhsm-client-jce
```

RHEL 8

```
$ sudo yum remove cloudhsm-client-jce
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client-jce
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client-jce
```

2. Pare o Client Daemon para o Client SDK 3.

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

3. Desinstale o Client Daemon do Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client
```

CentOS 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 8

```
$ sudo yum remove cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Note

As configurações personalizadas precisam ser habilitadas novamente.

4. Instale o provedor JCE do Client SDK seguindo as etapas em [Instale o provedor JCE para o AWS CloudHSM Client SDK 5](#).
5. O Client SDK 5 apresenta um novo formato de arquivo de configuração e uma ferramenta de inicialização de linha de comando. Para inicializar seu provedor JCE do Client SDK 5, siga as instruções listadas no guia do usuário em [Bootstrap o Client SDK](#).
6. Teste sua aplicação em seu ambiente de desenvolvimento. Faça atualizações no código existente para resolver as alterações importantes antes da migração final.

Tópicos relacionados

- [Melhores práticas para AWS CloudHSM](#)

Usando o Client SDK 5 para trabalhar com AWS CloudHSM

AWS CloudHSM inclui duas versões principais do SDK do cliente:

- Client SDK 5: este é o nosso SDK de cliente padrão e mais recente. Para obter informações sobre os benefícios e vantagens que ela oferece, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

- Client SDK 3: este é o nosso SDK do cliente mais antigo. Ele inclui um conjunto completo de componentes para ferramentas de gerenciamento e compatibilidade de aplicativos baseados em plataformas e linguagens.

Para obter instruções sobre como migrar do Client SDK 3 para o Client SDK 5, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Este tópico descreve o Client SDK 5. Para verificar qual versão do Client SDK você está usando, consulte

Tópicos

- [Benefícios do AWS CloudHSM Client SDK 5](#)
- [AWS CloudHSM Plataformas compatíveis com o Client SDK 5](#)
- [Biblioteca PKCS #11 para AWS CloudHSM Client SDK 5](#)
- [Mecanismo AWS CloudHSM dinâmico OpenSSL para cliente SDK 5](#)
- [Provedor AWS CloudHSM de OpenSSL para cliente SDK 5](#)
- [Provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#)
- [Provedor de JCE para AWS CloudHSM Client SDK 5](#)

Benefícios do AWS CloudHSM Client SDK 5

Comparado ao AWS CloudHSM Client SDK 3, o Client SDK 5 é mais fácil de gerenciar, oferece maior configurabilidade e maior confiabilidade. O Client SDK 5 também oferece algumas vantagens importantes além das proporcionadas pelo Client SDK 3.

Projetado para arquitetura com tecnologia sem servidor

O Client SDK 5 não exige um daemon de cliente, então você não precisa mais gerenciar um serviço em segundo plano. Isso ajuda os usuários de algumas maneiras importantes:

- Simplifica o processo de startup do aplicativo. Tudo o que você precisa fazer para começar a usar o CloudHSM é configurar o SDK antes de executar seu aplicativo.
- Você não precisa de um processo em execução constante, o que facilita a integração com componentes com tecnologia sem servidor, como Lambda e Elastic Container Service (ECS – serviço de contêiner elástico).

Melhores integrações de terceiros e portabilidade simplificada

O Client SDK 5 segue de perto a especificação JCE e fornece portabilidade mais fácil entre diferentes provedores de JCE e melhores integrações de terceiros

Experiência de usuário e capacidade de configuração aprimoradas

O Client SDK 5 melhora a legibilidade das mensagens de log e fornece exceções e mecanismos de tratamento de erros mais claros, o que torna a triagem de autoatendimento muito mais fácil para os usuários. O SDK 5 também oferece uma variedade de configurações, que estão listadas na [página Configurar ferramenta](#).

Suporte mais amplo à plataforma

O Client SDK 5 oferece mais suporte para plataformas operacionais modernas. Isso inclui suporte para tecnologias ARM e maior compatibilidade com [JCE](#), [PKCS #11](#) e [OpenSSL](#). Para obter mais informações, consulte [Plataformas compatíveis](#).

IPv6 suporte de conexão

O Client SDK 5.14+ suporta conexões com o uso de pilha dupla. HSMs IPv6

Atributos e mecanismos adicionais

O Client SDK 5 inclui atributos e mecanismos adicionais que não estão disponíveis no Client SDK 3 e o Client SDK 5 continuará adicionando mais mecanismos no futuro.

AWS CloudHSM Plataformas compatíveis com o Client SDK 5

O suporte básico é diferente para cada versão do SDK AWS CloudHSM do cliente. O suporte de plataforma para componentes em um SDK normalmente, mas nem sempre, corresponde ao suporte básico. Para determinar o suporte à plataforma para um componente específico, primeiro certifique-se de que a plataforma desejada apareça na seção base do SDK e, em seguida, verifique se há exclusões ou qualquer outra informação pertinente na seção de componentes.

AWS CloudHSM suporta somente sistemas operacionais de 64 bits.

O suporte da plataforma muda com o tempo. Versões anteriores do Client SDK do CloudHSM podem não ser compatíveis com todos os sistemas operacionais listados aqui. Use as notas de versão para determinar o suporte do sistema operacional para versões anteriores do Client SDK do CloudHSM. Para obter mais informações, consulte [Downloads para o SDK AWS CloudHSM do cliente](#).

Para ver as plataformas compatíveis com o Client SDK anterior, consulte [AWS CloudHSM Plataformas compatíveis com o Client SDK 3](#)

O Client SDK 5 não exige um daemon de cliente.

Tópicos

- [Suporte Linux para o AWS CloudHSM Client SDK 5](#)
- [Suporte do Windows para o AWS CloudHSM Client SDK 5](#)
- [Suporte sem servidor para o AWS CloudHSM Client SDK 5](#)
- [Compatibilidade com HSM para o AWS CloudHSM Client SDK 5](#)

Suporte Linux para o AWS CloudHSM Client SDK 5

AWS CloudHSM O Client SDK 5 é compatível com os seguintes sistemas operacionais e plataformas Linux.

Plataformas compatíveis	Arquitetura X86_64	Arquitetura ARM
Amazon Linux 2	Sim	Sim
Amazon Linux 2023	Sim	Sim
Red Hat Enterprise Linux 8 (8.3+)	Sim	Sim
Red Hat Enterprise Linux 9 (9.2+)	Sim	Sim
Red Hat Enterprise Linux 10 (10.0+)	Sim	Sim
Ubuntu 22.04 LTS	Sim	Sim
Ubuntu 24.04 LTS	Sim	Sim

- O SDK 5.16 foi a última versão a fornecer suporte à plataforma Ubuntu 20.04 LTS. Para obter mais informações, consulte o [site do Ubuntu](#).
- O SDK 5.12 foi a última versão a fornecer suporte à plataforma CentOS 7 (7.8+). Para obter mais informações, consulte o [site do CentOS](#).
- O SDK 5.12 foi a última versão a fornecer suporte à plataforma Red Hat Enterprise Linux 7 (7.8+). Para obter mais informações, acesse o [site do Red Hat](#).
- O SDK 5.4.2 foi a última versão a fornecer suporte à plataforma CentOS 8. Para obter mais informações, consulte o [site do CentOS](#).

Suporte do Windows para o AWS CloudHSM Client SDK 5

AWS CloudHSM O Client SDK 5 oferece suporte às seguintes versões do Windows Server.

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019
- Microsoft Windows Server 2022
- Microsoft Windows Server 2025

Suporte sem servidor para o AWS CloudHSM Client SDK 5

AWS CloudHSM O Client SDK 5 oferece suporte aos seguintes AWS serviços sem servidor.

- AWS Lambda
- Docker/ECS

Compatibilidade com HSM para o AWS CloudHSM Client SDK 5

A tabela a seguir descreve a compatibilidade AWS CloudHSM do Client SDK 5 para HSMs.

hsm1.medium	hsm2m.medium
Compatível com o Client SDK versão 5.0.0 e posteriores.	Compatível com o SDK do cliente versão 5.9.0 e posterior.

Biblioteca PKCS #11 para AWS CloudHSM Client SDK 5

O PKCS #11 é um padrão para realizar operações criptográficas em módulos de segurança de hardware (HSMs). AWS CloudHSM oferece implementações da biblioteca PKCS #11 que são compatíveis com a versão 2.40 do PKCS #11.

Para obter informações sobre bootstrapping, consulte [Conectar-se ao cluster](#). Para solução de problemas, consulte [Problemas conhecidos da biblioteca PKCS #11 para AWS CloudHSM](#).

Para obter mais informações sobre como usar Client SDK 3, consulte [Usando a versão anterior do SDK para trabalhar com AWS CloudHSM](#).

Tópicos

- [Instale a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Autentique-se na biblioteca PKCS #11 do AWS CloudHSM Client SDK 5](#)
- [Tipos de chave compatíveis com a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Mecanismos compatíveis com a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Operações de API suportadas para a biblioteca PKCS #11 para AWS CloudHSM Client SDK 5](#)
- [Atributos principais na biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Exemplos de código para a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Configurações avançadas para a biblioteca PKCS #11 para AWS CloudHSM](#)
- [Armazenamento de certificados com a biblioteca PKCS #11](#)

Instale a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

Este tópico fornece instruções para instalar a versão mais recente da biblioteca PKCS #11 para a série de versões do Client SDK 5 do AWS CloudHSM. Para obter mais informações sobre a biblioteca Client SDK ou do PKCS #11, consulte [Usando a biblioteca Client SDK](#) e [PKCS #11](#).

Com o Client SDK 5, você não precisa instalar ou executar um daemon de cliente.

Para executar um único cluster HSM com o Client SDK 5, você deve primeiro gerenciar as configurações de durabilidade da chave do cliente configurando `disable_key_availability_check` como `True`. Para obter mais informações, consulte [Sincronização de chave](#) e a [ferramenta de configuração do Client SDK 5](#).

Para obter mais informações sobre a biblioteca PKCS #11SDK do Client SDK 5, consulte [biblioteca PKCS #11](#).

 Note

Para executar um único cluster HSM com o Client SDK 5, você deve primeiro gerenciar as configurações de durabilidade da chave do cliente configurando `disable_key_availability_check` como `True`. Para obter mais informações, consulte [Sincronização de chave](#) e a [ferramenta de configuração do Client SDK 5](#).

Para instalar e configurar a biblioteca PKCS #11

1. Use os seguintes comandos para fazer download e instalar a biblioteca PKCS #11.

Amazon Linux 2023

Instale a biblioteca PKCS #11 para o Amazon Linux 2023 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-pkcs11-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.amzn2023.x86_64.rpm
```

Instale a biblioteca PKCS #11 para Amazon Linux 2023 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-pkcs11-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Instale a biblioteca PKCS #11 para Amazon Linux 2 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-pkcs11-latest.el7.x86_64.rpm
```



```
$ sudo yum install ./cloudhsm-pkcs11-latest.el7.x86_64.rpm
```

Instale a biblioteca PKCS #11 para Amazon Linux 2 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-pkcs11-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

Instale a biblioteca PKCS #11 para RHEL 10 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-pkcs11-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el10.x86_64.rpm
```

Instale a biblioteca PKCS #11 para RHEL 10 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-pkcs11-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

Instale a biblioteca PKCS #11 para RHEL 9 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-pkcs11-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el9.x86_64.rpm
```

Instale a biblioteca PKCS #11 para RHEL 9 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-pkcs11-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

Instale a biblioteca PKCS #11 para RHEL 8 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-pkcs11-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el8.x86_64.rpm
```

Instale a biblioteca PKCS #11 para RHEL 8 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-pkcs11-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el8.aarch64.rpm
```

Ubuntu 24.04 LTS

Instale a biblioteca PKCS #11 para Ubuntu 24.04 LTS na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-pkcs11_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u24.04_amd64.deb
```

Instale a biblioteca PKCS #11 para Ubuntu 24.04 LTS na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-pkcs11_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Instale a biblioteca PKCS #11 para Ubuntu 22.04 LTS na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-pkcs11_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u22.04_amd64.deb
```

Instale a biblioteca PKCS #11 para o Ubuntu 22.04 LTS na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-pkcs11_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u22.04_arm64.deb
```

Windows Server

Instale a biblioteca PKCS #11 para Windows Server na arquitetura X86_64:

1. Faça o download da [biblioteca PKCS #11 para Client SDK 5](#).
2. Execute o instalador da biblioteca PKCS #11 (AWSCloudHSMPCSKS11-latest.msi) com privilégios administrativos do Windows.
2. Use a ferramenta de configuração para especificar um local para o certificado de emissão. Para instruções, consulte [Especifique o local do certificado de emissão](#).
3. Para se conectar e usar o cluster [Bootstrap o Client SDK](#).
4. Você pode encontrar os arquivos da biblioteca do PKCS #11 nos seguintes locais:
 - Binários, scripts de configuração e arquivos de log do Linux:

```
/opt/cloudhsm
```

Binários do Windows:

```
C:\Program Files\Amazon\CloudHSM
```

Scripts de configuração e arquivos de log do Windows:

```
C:\ProgramData\Amazon\CloudHSM
```

Autentique-se na biblioteca PKCS #11 do AWS CloudHSM Client SDK 5

Quando você usa a biblioteca PKCS #11, seu aplicativo é executado como um [usuário criptográfico \(UC\)](#) específico em seu HSMs in. AWS CloudHSM O aplicativo pode visualizar e gerenciar apenas as chaves que o CU possui e compartilha. Você pode usar uma UC existente em seu aplicativo HSMs ou criar uma nova UC para seu aplicativo. Para obter informações sobre gerenciamento CUs, consulte [Gerenciando usuários do HSM com o CloudHSM CLI e Gerenciando usuários do HSM com o CloudHSM Management Utility \(CMU\)](#)

Para especificar o CU para a biblioteca PKCS #11, use o parâmetro do PIN da [função C_Login](#) do PKCS #11. Para AWS CloudHSM, o parâmetro pin tem o seguinte formato:

```
<CU_user_name>:<password>
```

Por exemplo, o seguinte comando define o PIN de PKCS #11 como o CU com nome de usuário CryptoUser e senha CUPassword123!.

```
CryptoUser:CUPassword123!
```

Tipos de chave compatíveis com a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

A biblioteca PKCS #11 para AWS CloudHSM Client SDK 5 é compatível com os seguintes tipos de chave.

Tipo de chave	Description
AES	Gere chaves AES de 128, 192 e 256 bits.

Tipo de chave	Description
DES triplo (3DES, DESede)	Gere chaves DES triplas de 192 bits. Consulte a nota 1 abaixo para ver uma mudança futura.
EC	Gera chaves com as curvas secp224r1 (P-224), secp256r1 (P-256), secp256k1 (Blockchain), secp384r1 (P-384) e secp521r1 (P-521).
GENERIC_SECRET	Gerar chaves genéricas de 1 a 800 bytes.
RSA	Gera chaves RSA de 2.048 a 4.096 bits, em incrementos de 256 bits.

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Mecanismos compatíveis com a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

A biblioteca PKCS #11 é compatível com a versão 2.40 da especificação PKCS #11. Para invocar um recurso de criptografia usando o PKCS#11, chame uma função com um determinado mecanismo. As seções a seguir resumem as combinações de funções e mecanismos compatíveis com o AWS CloudHSM Client SDK 5.

A biblioteca de software do para PKCS #11 oferece suporte aos seguintes algoritmos:

- Criptografia e descriptografia — AES-CBC, AES-CTR, AES-ECB, AES-GCM, -CBC, -ECB, RSA-OAEP e RSA-PKCS DES3 DES3
- Assinar e verificar: RSA, HMAC e ECDSA; com e sem hash
- Hash/digest — SHA1,, SHA224, e SHA256 SHA384 SHA512
- Encapsulamento de chave: AES Key Wrap¹, AES-GCM, RSA-AES e RSA-OAEP
- Derivação de chave — SP800 -108 Counter KDF e ECDH com KDF (os algoritmos KDF suportados são X9.63 com,,,) SHA1 SHA224 SHA256 SHA384 SHA512

Tópicos

- [Gere funções de chave e par de chaves](#)
- [Funções de assinatura e verificação](#)
- [Assine, recupere e verifique as funções de recuperação](#)
- [Funções de resumo](#)
- [Funções de criptografia e descriptografia](#)
- [Derivar funções de chave](#)
- [Funções de agrupamento e desagrupamento](#)
- [Tamanho máximo de dados para cada mecanismo](#)
- [Anotações do mecanismo](#)

Gere funções de chave e par de chaves

A biblioteca AWS CloudHSM de software da biblioteca PKCS #11 permite que você use os seguintes mecanismos para as funções Generate Key e Key Pair.

- CKM_RSA_PKCS_KEY_PAIR_GEN
- CKM_RSA_X9_31_KEY_PAIR_GEN: esse mecanismo é funcionalmente idêntico ao mecanismo CKM_RSA_PKCS_KEY_PAIR_GEN, mas oferece maiores garantias para geração de p e de q.
- CKM_EC_KEY_PAIR_GEN
- CKM_GENERIC_SECRET_KEY_GEN
- CKM_AES_KEY_GEN
- CKM_DES3_KEY_GEN: próxima mudança listada na nota de rodapé [5](#).

Funções de assinatura e verificação

A biblioteca AWS CloudHSM de software da biblioteca PKCS #11 permite que você use os seguintes mecanismos para as funções de assinatura e verificação. Com o Client SDK 5, os dados são codificados localmente no software. Isso significa que não há limite no tamanho dos dados que podem ser criptografados pelo SDK.

Com o Client SDK 5, o hashing RSA e ECDSA é feito localmente; portanto, não há limite de dados. Com o HMAC, há um limite de dados. Consulte a nota de rodapé [2](#) para obter mais informações.

RSA

- CKM_RSA_X_509
- CKM_RSA_PKCS: somente operações de uma única parte.
- CKM_RSA_PKCS_PSS: somente operações de uma única parte.
- CKM_SHA1_RSA_PKCS
- CKM_SHA224_RSA_PKCS
- CKM_SHA256_RSA_PKCS
- CKM_SHA384_RSA_PKCS
- CKM_SHA512_RSA_PKCS
- CKM_SHA512_RSA_PKCS
- CKM_SHA1_RSA_PKCS_PSS
- CKM_SHA224_RSA_PKCS_PSS
- CKM_SHA256_RSA_PKCS_PSS
- CKM_SHA384_RSA_PKCS_PSS
- CKM_SHA512_RSA_PKCS_PSS

ECDSA

- CKM_ECDSA: somente operações de uma única parte.
- CKM_ECDSA_SHA1
- CKM_ECDSA_SHA224
- CKM_ECDSA_SHA256
- CKM_ECDSA_SHA384
- CKM_ECDSA_SHA512

HMAC

- CKM_SHA_1_HMAC²
- CKM_SHA224_HMAC²
- CKM_SHA256_HMAC²

- CKM_SHA384_HMAC²
- CKM_SHA512_HMAC²

CMAC

- CKM_AES_CMAC

Assine, recupere e verifique as funções de recuperação

O Client SDK 5 não oferece suporte às funções Sign Recover (recuperação de assinatura) e Verify Recover (verificar recuperação).

Funções de resumo

A biblioteca AWS CloudHSM de software da biblioteca PKCS #11 permite que você use os seguintes mecanismos para funções Digest. Com o Client SDK 5, os dados são codificados localmente no software. Isso significa que não há limite no tamanho dos dados que podem ser criptografados pelo SDK.

- CKM_SHA_1
- CKM_SHA224
- CKM_SHA256
- CKM_SHA384
- CKM_SHA512

Funções de criptografia e descriptografia

A biblioteca AWS CloudHSM de software da biblioteca PKCS #11 permite que você use os seguintes mecanismos para as funções Encrypt e Decrypt.

- CKM_RSA_X_509
- CKM_RSA_PKCS: somente operações de uma única parte. Próxima mudança listada na nota de rodapé ⁵.
- CKM_RSA_PKCS_OAEP: somente operações de uma única parte.
- CKM_AES_ECB
- CKM_AES_CTR

- CKM_AES_CBC
- CKM_AES_CBC_PAD
- CKM_DES3_CBC: próxima mudança listada na nota de rodapé [5](#).
- CKM_DES3_ECB: próxima mudança listada na nota de rodapé [5](#).
- CKM_DES3_CBC_PAD: próxima mudança listada na nota de rodapé [5](#).
- CKM_AES_GCM [1](#), [2](#)
- CKM_CLOUDHSM_AES_GCM [3](#)

Derivar funções de chave

A biblioteca AWS CloudHSM de software para a biblioteca PKCS #11 suporta os seguintes mecanismos de derivação de chaves:

- CKM_SP800_108_COUNTER_KDF
- CKM_ECDH1_DERIVE: é compatível com derivação de chave ECDH com os seguintes tipos de KDF definidos pelo fornecedor: [6](#)
 - CKD_CLOUDHSM_X963_SHA1_KDF- X9.63 KDF com SHA1 [7](#)
 - CKD_CLOUDHSM_X963_SHA224_KDF- X9.63 KDF com SHA224 [7](#)
 - CKD_CLOUDHSM_X963_SHA256_KDF- X9.63 KDF com SHA256 [7](#)
 - CKD_CLOUDHSM_X963_SHA384_KDF- X9.63 KDF com SHA384 [7](#)
 - CKD_CLOUDHSM_X963_SHA512_KDF- X9.63 KDF com SHA512 [7](#)

Funções de agrupamento e desagrupamento

A biblioteca AWS CloudHSM de software da biblioteca PKCS #11 permite que você use os seguintes mecanismos para as funções Wrap e Unwrap.

Para obter opções adicionais de empacotamento de chaves AES, consulte [Empacotamento de chaves AES](#).

- CKM_RSA_PKCS: somente operações de uma única parte. Uma nova mudança futura listada na nota de rodapé [5](#).
- CKM_RSA_PKCS_OAEP [4](#)

- CKM_AES_GCM^{1, 3}
- CKM_CLOUDHSM_AES_GCM³
- CKM_RSA_AES_KEY_WRAP
- CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD³
- CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD³
- CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD³

Tamanho máximo de dados para cada mecanismo

A tabela a seguir lista o tamanho máximo de dados definido para cada mecanismo:

Tamanho máximo do conjunto de dados

Mecanismo	Tamanho máximo de dados em bytes
CKM_SHA_1_HMAC	16288
CKM_SHA224_HMAC	16256
CKM_SHA256_HMAC	16288
CKM_SHA384_HMAC	16224
CKM_SHA512_HMAC	16224
CKM_AES_CBC	16272
CKM_AES_GCM	16224
CKM_CLOUDHSM_AES_GCM	16224
CKM_DES3_CBC	16280

Anotações do mecanismo

- [1] Ao executar a criptografia AES-GCM, o HSM não aceitará dados do vetor de inicialização (IV) do aplicativo. Você deve usar um IV gerado por ele. O IV de 12 bytes fornecido pelo HSM

é gravado na referência da memória apontada pelo elemento pIV da estrutura de parâmetros CK_GCM_PARAMS que você fornece. Para que não haja confusão para o usuário, o SDK do PKCS #11 na versão 1.1.1 e posterior, garante que o pIV aponte para um buffer zerado quando a criptografia AES-GCM é inicializada.

- [2] Ao trabalhar com dados usando qualquer um dos mecanismos a seguir, se o buffer de dados exceder o tamanho máximo de dados, a operação resultará em um erro. Para esses mecanismos, todo o processamento de dados deve ocorrer dentro do HSM. Para obter informações sobre conjuntos de tamanho máximo de dados para cada mecanismo, consulte [Tamanho máximo de dados para cada mecanismo](#).
- [3] Mecanismo definido pelo fornecedor. Para usar os mecanismos definidos pelo fornecedor do CloudHSM, os aplicativos PKCS #11 devem incluir /opt/cloudhsm/include/pkcs11/pkcs11t.h durante a compilação.

CKM_CLOUDHSM_AES_GCM: Este mecanismo proprietário é uma alternativa programaticamente mais segura para o padrão CKM_AES_GCM. Ele antecede o IV gerado pelo HSM para o texto cifrado em vez de escrevê-lo de volta na estrutura CK_GCM_PARAMS fornecida durante a inicialização da cifra. Você pode usar esse mecanismo com as funções C_Encrypt, C_WrapKey, C_Decrypt, e C_UnwrapKey. Ao usar esse mecanismo, a variável pIV no struct CK_GCM_PARAMS deve ser definida como NULL. Ao usar este mecanismo com C_Decrypt e C_UnwrapKey, espera-se que o IV seja precedido pelo texto cifrado que está sendo desencapsulado.

CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD: Agrupamento de chaves AES com preenchimento PKCS #5.

CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD: Agrupamento de chaves AES com preenchimento de zeros.

- [4] Os seguintes CK_MECHANISM_TYPE e CK_RSA_PKCS_MGF_TYPE são compatíveis como CK_RSA_PKCS_OAEP_PARAMS para CKM_RSA_PKCS_OAEP:
 - CKM_SHA_1 usando CKG_MGF1_SHA1
 - CKM_SHA224 usando CKG_MGF1_SHA224
 - CKM_SHA256 usando CKG_MGF1_SHA256
 - CKM_SHA384 usando CKM_MGF1_SHA384
 - CKM_SHA512 usando CKM_MGF1_SHA512
- [5] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

- [6] Tipos definidos pelo fornecedor. Para usar os tipos definidos pelo fornecedor do CloudHSM, as aplicação PKCS #11 devem incluir `cloudhsm_pkcs11_vendor_defs.h` durante a compilação. Isso pode ser localizado em `/opt/cloudhsm/include/pkcs11/cloudhsm_pkcs11_vendor_defs.h` para plataformas baseadas em Linux e em `C:\Program Files\Amazon\CloudHSM\include\pkcs11\cloudhsm_pkcs11_vendor_defs.h` para plataformas baseadas no Windows
- [7] As principais funções de derivação (KDFs) estão especificadas na [Publicação Especial 800-56A do NIST](#), Revisão 3.

Operações de API suportadas para a biblioteca PKCS #11 para AWS CloudHSM Client SDK 5

A biblioteca PKCS #11 oferece suporte às seguintes operações de API PKCS #11 para o AWS CloudHSM Client SDK 5.

- `C_CloseAllSessions`
- `C_CloseSession`
- `C_CreateObject`
- `C_Decrypt`
- `C_DecryptFinal`
- `C_DecryptInit`
- `C_DecryptUpdate`
- `C_DeriveKey`
- `C_DestroyObject`
- `C_Digest`
- `C_DigestFinal`
- `C_DigestInit`
- `C_DigestUpdate`
- `C_Encrypt`
- `C_EncryptFinal`
- `C_EncryptInit`
- `C_EncryptUpdate`

- C_Finalize
- C_FindObjects
- C_FindObjectsFinal
- C_FindObjectsInit
- C_GenerateKey
- C_GenerateKeyPair
- C_GenerateRandom
- C_GetAttributeValue
- C_GetFunctionList
- C_GetInfo
- C_GetMechanismInfo
- C_GetMechanismList
- C_GetSessionInfo
- C_GetSlotInfo
- C_GetSlotList
- C_GetTokenInfo
- C_Initialize
- C_Login
- C_Logout
- C_OpenSession
- C_Sign
- C_SignFinal
- C_SignInit
- C_SignUpdate
- C_UnWrapKey
- C_Verify
- C_VerifyFinal
- C_VerifyInit
- C_VerifyUpdate

- C_WrapKey

Atributos principais na biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

Um objeto AWS CloudHSM chave pode ser uma chave pública, privada ou secreta. As ações permitidas em um objeto de chave são especificadas por meio de atributos. Os atributos são definidos quando o objeto de chave é criado. Quando você usa a biblioteca PKCS #11 para AWS CloudHSM, atribuímos valores padrão conforme especificado pelo padrão PKCS #11.

AWS CloudHSM não suporta todos os atributos listados na especificação PKCS #11. Estamos em conformidade com a especificação de todos os atributos aos quais oferecemos suporte. Esses atributos estão listados nas respectivas tabelas.

Funções criptográficas como C_CreateObject, C_GenerateKey, C_GenerateKeyPair, C_UnwrapKey, e C_DeriveKey que criam, modificam ou copiam objetos usam um modelo de atributo como um de seus parâmetros. Para obter mais informações sobre como passar um modelo de atributo durante a criação do objeto, consulte o exemplo para [Gerar chaves por meio da biblioteca PKCS #11](#).

Os tópicos a seguir fornecem mais informações sobre os AWS CloudHSM principais atributos.

Tópicos

- [Tabelas de atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Modificando atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)
- [Interpretação dos códigos de erro da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#)

Tabelas de atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

As tabelas da biblioteca PKCS #11 AWS CloudHSM contêm uma lista de atributos que diferem por tipo de chave. Ele indica se um determinado atributo é compatível com um determinado tipo de chave ao usar uma função criptográfica específica com AWS CloudHSM.

Legenda:

- ✓ indica que o CloudHSM oferece suporte ao atributo para o tipo de chave específico.
- ✕ indica que o CloudHSM não oferece suporte ao atributo para o tipo de chave específico.
- R indica que o valor do atributo é definido como somente leitura para o tipo de chave específico.
- S indica que o atributo não pode ser lido pelo GetAttributeValue pois ele é confidencial.

- Uma célula vazia na coluna Valor padrão indica que não há nenhum valor padrão específico atribuído ao atributo.

GenerateKeyPair

Atributo	Tipo de chave				Valor padrão
	EC privada	EC pública	RSA privada	RSA pública	
CKA_CLASS	✓	✓	✓	✓	
CKA_KEY_TYPE	✓	✓	✓	✓	
CKA_LABEL	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	Verdadeiro
CKA_TOKEN	✓	✓	✓	✓	Falso
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT	✗	✓	✗	✓	Falso
CKA_DECRYPT	✓	✗	✓	✗	Falso
CKA_DERIVE	✓	✓	✓	✓	Falso

Atributo	Tipo de chave				Valor padrão
CKA_MODIFIABLE	✓	✓	✓	✓	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✗	✓	✗	Falso
CKA_SIGN_RECOVER	✗	✗	✗	✗	
CKA_VERIFY	✗	✓	✗	✓	Falso
CKA_VERIFY_RECOVER	✗	✗	✗	✗	
CKA_WRAP	✗	✓	✗	✓	Falso
CKA_WRAP_TEMPLATE	✗	✓	✗	✓	
CKA_TRUSTED	✗	✓	✗	✓	Falso
CKA_WRAP_WITH_TRUSTED	✓	✗	✓	✗	Falso
CKA_UNWRAP	✓	✗	✓	✗	Falso
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	

Atributo	Tipo de chave				Valor padrão
CKA_SENSITIVE	✓ ¹	✗	✓ ¹	✗	Verdadeiro
CKA_ALWAYS_SENSITIVE	R	✗	R	✗	
CKA_EXTRACTABLE	✓	✗	✓	✗	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	✗	R	✗	
CKA_MODULUS	✗	✗	✗	✗	
CKA_MODULUS_BITS	✗	✗	✗	✓ ²	
CKA_PRIME_1	✗	✗	✗	✗	
CKA_PRIME_2	✗	✗	✗	✗	
CKA_COEFFICIENT	✗	✗	✗	✗	
CKA_EXPONENT_1	✗	✗	✗	✗	
CKA_EXPONENT_2	✗	✗	✗	✗	

Atributo	Tipo de chave				Valor padrão
CKA_PRIVATE_EXPONENT	×	×	×	×	
CKA_PUBLIC_EXPONENT	×	×	×	✓ ²	
CKA_EC_PARAMS	×	✓ ²	×	×	
CKA_EC_POINT	×	×	×	×	
CKA_VALUE	×	×	×	×	
CKA_VALUE_LEN	×	×	×	×	
CKA_CHECK_VALUE	R	R	R	R	

GenerateKey

Atributo	Tipo de chave			Valor padrão
	AES	DES3	Segredo genérico	
CKA_CLASS	✓	✓	✓	
CKA_KEY_TYPE	✓	✓	✓	

Atributo	Tipo de chave			Valor padrão
CKA_LABEL		✓	✓	✓
CKA_ID		✓	✓	✓
CKA_LOCAL		R	R	R
CKA_TOKEN		✓	✓	✓
CKA_PRIVATE		✓ ¹	✓ ¹	✓ ¹
CKA_ENCRYPT		✓	✓	✗
CKA_DECRYPT		✓	✓	✗
CKA_DERIVE		✓	✓	✓
CKA_MODIFIABLE		✓	✓	✓
CKA_DESTROYABLE		✓	✓	✓
CKA_SIGN		✓	✓	✓
CKA_SIGN_RECOVER		✗	✗	✗
CKA_VERIFY		✓	✓	✓
CKA_VERIFY_RECOVER		✗	✗	✗

Atributo	Tipo de chave				Valor padrão
CKA_WRAP		✓	✓	✗	Falso
CKA_WRAP_TEMPLATE		✓	✓	✗	
CKA_TRUSTED		✓	✓	✗	Falso
CKA_WRAP_WITH_TRUSTED		✓	✓	✓	Falso
CKA_UNWRAP		✓	✓	✗	Falso
CKA_UNWRAP_TEMPLATE		✓	✓	✗	
CKA_SENSITIVE		✓	✓	✓	Verdadeiro
CKA_ALWAYS_SENSITIVE		✗	✗	✗	
CKA_EXTRACTABLE		✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE		R	R	R	
CKA_MODULUS		✗	✗	✗	

Atributo	Tipo de chave			Valor padrão
CKA_MODULUS_BITS	×	×	×	
CKA_PRIME_1	×	×	×	
CKA_PRIME_2	×	×	×	
CKA_COEFFICIENT	×	×	×	
CKA_EXPONENT_1	×	×	×	
CKA_EXPONENT_2	×	×	×	
CKA_PRIVATE_EXPONENT	×	×	×	
CKA_PUBLIC_EXPONENT	×	×	×	
CKA_EC_PARAMS	×	×	×	
CKA_EC_POINT	×	×	×	
CKA_VALUE	×	×	×	
CKA_VALUE_LEN	✓ ²	×	✓ ²	

Atributo	Tipo de chave				Valor padrão
CKA_CHECK_VALUE		R	R	R	

CreateObject

Atributo	Tipo de chave							Valor padrão
	EC privada	EC pública	RSA privada	RSA pública	AES	DES3	Segredo genérico	
CKA_CLASS	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	
CKA_KEY_TYPE	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	R	R	R	Falso
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓	Falso
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT	✗	✗	✗	✓	✓	✓	✗	Falso
CKA_DECRYPT	✗	✗	✓	✗	✓	✓	✗	Falso

Atributo	Tipo de chave							Valor padrão
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	Falso
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✗	✓	✗	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✗	✗	✗	✗	✗	✗	✗	Falso
CKA_VERIFY	✗	✓	✗	✓	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✗	✗	✗	✗	✗	✗	✗	
CKA_WRAP	✗	✗	✗	✓	✓	✓	✗	Falso
CKA_WRAP_TEMPLATE	✗	✓	✗	✓	✓	✓	✗	
CKA_TRUSTED	✗	✓	✗	✓	✓	✓	✗	Falso
CKA_WRAP_WITH_TRUSTED	✓	✗	✓	✗	✓	✓	✓	Falso
CKA_UNWRAP	✗	✗	✓	✗	✓	✓	✗	Falso

Atributo	Tipo de chave							Valor padrão
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	✓	✓	✗	
CKA_SENSITIVE	✓	✗	✓	✗	✓	✓	✓	Verdadeiro
CKA_ALWAYS_SENSITIVE	R	✗	R	✗	R	R	R	
CKA_EXTRACTABLE	✓	✗	✓	✗	✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	✗	R	✗	R	R	R	
CKA_MODULUS	✗	✗	✓ ²	✓ ²	✗	✗	✗	
CKA_MODULUS_BITS	✗	✗	✗	✗	✗	✗	✗	
CKA_PRIME_1	✗	✗	✓	✗	✗	✗	✗	
CKA_PRIME_2	✗	✗	✓	✗	✗	✗	✗	
CKA_COEFFICIENT	✗	✗	✓	✗	✗	✗	✗	
CKA_EXPONENT_1	✗	✗	✓	✗	✗	✗	✗	

Atributo	Tipo de chave							Valor padrão
CKA_EXPONENT_2	×	×	✓	×	×	×	×	
CKA_PRIVATE_EXPONENT	×	×	✓ ²	×	×	×	×	
CKA_PUBLIC_EXPONENT	×	×	✓ ²	✓ ²	×	×	×	
CKA_EC_PARAMS	✓ ²	✓ ²	×	×	×	×	×	
CKA_EC_POINT	×	✓ ²	×	×	×	×	×	
CKA_VALUE	✓ ²	×	×	×	✓ ²	✓ ²	✓ ²	
CKA_VALUE_LEN	×	×	×	×	×	×	×	
CKA_CHECK_VALUE	R	R	R	R	R	R	R	

UnwrapKey

Atributo	Tipo de chave						Valor padrão
		EC privada	RSA privada	AES	DES3	Segredo genérico	

Atributo	Tipo de chave						Valor padrão
CKA_CLASS		✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	
CKA_KEY_TYPE		✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	
CKA_LABEL		✓	✓	✓	✓	✓	
CKA_ID		✓	✓	✓	✓	✓	
CKA_LOCAL		R	R	R	R	R	Falso
CKA_TOKEN		✓	✓	✓	✓	✓	Falso
CKA_PRIVATE		✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT		✗	✗	✓	✓	✗	Falso
CKA_DECRYPT		✗	✓	✓	✓	✗	Falso
CKA_DERIVE		✓	✓	✓	✓	✓	Falso
CKA_MODIFIABLE		✓	✓	✓	✓	✓	Verdadeiro
CKA_DESTROYABLE		✓	✓	✓	✓	✓	Verdadeiro
CKA_SIGN		✓	✓	✓	✓	✓	Falso

Atributo	Tipo de chave						Valor padrão
CKA_SIGN_RECOVER		×	×	×	×	×	Falso
CKA_VERIFY		×	×	✓	✓	✓	Falso
CKA_VERIFY_RECOVER		×	×	×	×	×	
CKA_WRAP		×	×	✓	✓	×	Falso
CKA_UNWRAP		×	✓	✓	✓	×	Falso
CKA_SENSITIVE		✓	✓	✓	✓	✓	Verdadeiro
CKA_EXTRACTABLE		✓	✓	✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE		R	R	R	R	R	
CKA_ALWAYS_SENSITIVE		R	R	R	R	R	
CKA_MODULUS		×	×	×	×	×	
CKA_MODULUS_BITS		×	×	×	×	×	

Atributo	Tipo de chave						Valor padrão
CKA_PRIME_1		×	×	×	×	×	
CKA_PRIME_2		×	×	×	×	×	
CKA_COEFFICIENT		×	×	×	×	×	
CKA_EXPONENT_1		×	×	×	×	×	
CKA_EXPONENT_2		×	×	×	×	×	
CKA_PRIVATE_EXPONENT		×	×	×	×	×	
CKA_PUBLIC_EXPONENT		×	×	×	×	×	
CKA_EC_PARAMS		×	×	×	×	×	
CKA_EC_POINT		×	×	×	×	×	
CKA_VALUE		×	×	×	×	×	
CKA_VALUE_LEN		×	×	×	×	×	

Atributo	Tipo de chave						Valor padrão
CKA_CHECK_VALUE		R	R	R	R	R	

DeriveKey

Atributo	Tipo de chave				Valor padrão
		AES	DES3	Segredo genérico	
CKA_CLASS		✓ ²	✓ ²	✓ ²	
CKA_KEY_TYPE		✓ ²	✓ ²	✓ ²	
CKA_LABEL		✓	✓	✓	
CKA_ID		✓	✓	✓	
CKA_LOCAL		R	R	R	Verdadeiro
CKA_TOKEN		✓	✓	✓	Falso
CKA_PRIVATE		✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT		✓	✓	✗	Falso
CKA_DECRYPT		✓	✓	✗	Falso
CKA_DERIVE		✓	✓	✓	Falso

Atributo	Tipo de chave				Valor padrão
CKA_MODIFIABLE		✓	✓	✓	Verdadeiro
CKA_DESTROYABLE		✓	✓	✓	Verdadeiro
CKA_SIGN		✓	✓	✓	Falso
CKA_SIGN_RECOVER		✗	✗	✗	
CKA_VERIFY		✓	✓	✓	Falso
CKA_VERIFY_RECOVER		✗	✗	✗	
CKA_WRAP		✓	✓	✗	Falso
CKA_UNWRAP		✓	✓	✗	Falso
CKA_SENSITIVE		R	R	R	Verdadeiro
CKA_EXTRACTABLE		✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE		R	R	R	
CKA_ALWAYS_SENSITIVE		R	R	R	

Atributo	Tipo de chave			Valor padrão
CKA_MODULUS		x	x	x
CKA_MODULUS_BITS		x	x	x
CKA_PRIME_1		x	x	x
CKA_PRIME_2		x	x	x
CKA_COEFFICIENT		x	x	x
CKA_EXPONENT_1		x	x	x
CKA_EXPONENT_2		x	x	x
CKA_PRIVATE_EXPONENT		x	x	x
CKA_PUBLIC_EXPONENT		x	x	x
CKA_EC_PARAMS		x	x	x
CKA_EC_POINT		x	x	x
CKA_VALUE		x	x	x

Atributo	Tipo de chave			Valor padrão
CKA_VALUE_LEN		✓ ²	✗	✓ ²
CKA_CHECK_VALUE	R	R	R	

GetAttributeValue

Atributo	Tipo de chave						
	EC privada	EC pública	RSA privada	RSA pública	AES	DES3	Segredo genérico
CKA_CLASS	✓	✓	✓	✓	✓	✓	✓
CKA_KEY_TYPE	✓	✓	✓	✓	✓	✓	✓
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓
CKA_ID	✓	✓	✓	✓	✓	✓	✓
CKA_LOCAL	✓	✓	✓	✓	✓	✓	✓
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹
CKA_ENCRYPT	✗	✗	✗	✓	✓	✓	✗

Atributo	Tipo de chave						
CKA_DECRYPT	×	×	✓	×	✓	✓	×
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	×	✓	×	✓	✓	✓
CKA_SIGN_RECOVER	×	×	✓	×	×	×	×
CKA_VERIFY	×	✓	×	✓	✓	✓	✓
CKA_VERIFY_RECOVER	×	×	×	✓	×	×	×
CKA_WRAP	×	×	×	✓	✓	✓	×
CKA_WRAP_TEMPLATE	×	✓	×	✓	✓	✓	×
CKA_TRUSTED	×	✓	×	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	×	✓	×	✓	✓	✓
CKA_UNWRAP	×	×	✓	×	✓	✓	×

Atributo	Tipo de chave						
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	✓	✓	✗
CKA_SENSITIVE	✓	✗	✓	✗	✓	✓	✓
CKA_EXTRACTABLE	✓	✗	✓	✗	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✗	✓	✗	✓	✓	✓
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	R	R
CKA_MODULUS	✗	✗	✓	✓	✗	✗	✗
CKA_MODULUS_BITS	✗	✗	✗	✓	✗	✗	✗
CKA_PRIME_1	✗	✗	S	✗	✗	✗	✗
CKA_PRIME_2	✗	✗	S	✗	✗	✗	✗
CKA_COEFFICIENT	✗	✗	S	✗	✗	✗	✗
CKA_EXPONENT_1	✗	✗	S	✗	✗	✗	✗

Atributo	Tipo de chave						
CKA_EXPONENT_2	×	×	S	×	×	×	×
CKA_PRIVATE_EXPONENT	×	×	S	×	×	×	×
CKA_PUBLIC_EXPONENT	×	×	✓	✓	×	×	×
CKA_EC_PARAMS	✓	✓	×	×	×	×	×
CKA_EC_POINT	×	✓	×	×	×	×	×
CKA_VALUE	S	×	×	×	✓	✓	✓
CKA_VALUE_LEN	×	×	×	×	✓	×	✓
CKA_CHECK_VALUE	✓	✓	✓	✓	✓	✓	×

Anotações de atributos

- [1] Este atributo tem suporte parcial do firmware e deve ser explicitamente definido apenas como o valor padrão.
- [2] Atributo obrigatório.

Modificando atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

Alguns atributos da biblioteca PKCS #11 de um AWS CloudHSM objeto podem ser modificados após a criação do objeto, enquanto outros não. Para modificar atributos, use o comando [key set-attribute](#)

da CloudHSM CLI. Você também pode derivar uma lista de atributos usando o comando [key list](#) da CloudHSM CLI.

A lista a seguir exibe os atributos cuja modificação é permitida após a criação do objeto:

- CKA_LABEL
- CKA_TOKEN

 Note

A modificação é permitida somente para alterar uma chave de sessão para uma chave de token. Use o comando [key set-attribute](#) da CloudHSM CLI para alterar o valor do atributo.

- CKA_ENCRYPT
- CKA_DECRYPT
- CKA_SIGN
- CKA_VERIFY
- CKA_WRAP
- CKA_UNWRAP
- CKA_LABEL
- CKA_SENSITIVE
- CKA_DERIVE

 Note

Esse atributo oferece suporte à derivação de chaves. Ele deve ser False para todas as chaves públicas e não pode ser definido como True. Para chaves privadas EC e secretas, ele pode ser definido como True ou False.

- CKA_TRUSTED

 Note

Esse atributo pode ser definido como True ou False somente pelo Responsável pela criptografia (CO)

- CKA_WRAP_WITH_TRUSTED

 Note

Aplice esse atributo a uma chave de dados exportável para especificar que você só pode agrupar essa chave com chaves marcadas como CKA_TRUSTED. Depois que CKA_WRAP_WITH_TRUSTED for definido como verdadeiro, o atributo se torna somente para leitura e não é possível alterar ou remover o atributo.

Interpretação dos códigos de erro da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

Especificar no modelo um atributo da biblioteca PKCS #11 que não tenha suporte de uma chave específica resultará em um erro. A tabela a seguir contém códigos de erro que são gerados quando as especificações são violadas:

Código de erro	Descrição
CKR_TEMPLATE_INCONSISTENT	Você receberá esse erro quando especificar um atributo no modelo de atributo, em que o atributo está em conformidade com a especificação do PKCS #11, mas não tem suporte do CloudHSM.
CKR_ATTRIBUTE_TYPE_INVALID	Você receberá esse erro quando recuperar o valor de um atributo, que está em conformidade com a especificação do PKCS #11, mas não tem suporte do CloudHSM.
CKR_ATTRIBUTE_INCOMPLETE	Você receberá esse erro quando não especificar o atributo obrigatório no modelo de atributo.
CKR_ATTRIBUTE_READ_ONLY	Você receberá esse erro quando especificar um atributo somente leitura no modelo de atributo.

Exemplos de código para a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5

Os exemplos de código mostrados no GitHub mostram como realizar tarefas básicas usando a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5.

Pré-requisitos

Antes de executar os exemplos, realize as seguintes etapas para configurar o ambiente:

- Instale e configure a [biblioteca PKCS #11](#) para o Client SDK 5.
- Configure um [usuário de criptografia \(CU\)](#). Seu aplicativo usa essa conta do HSM para executar as amostras de código no HSM.

Exemplos de código

Exemplos de código para a biblioteca AWS CloudHSM de software para PKCS #11 estão disponíveis em [GitHub](#). Este repositório inclui exemplos de como fazer operações comuns usando PKCS #11, incluindo criptografia, descriptografia, assinatura e verificação.

- [Gerar chaves \(AES, RSA, EC\)](#)
- [Listar atributos de chave](#)
- [Criptografar e descriptografar dados com AES-GCM](#)
- [Encrypt and decrypt data with AES_CTR](#)
- [Encrypt and decrypt data with 3DES](#)
- [Assinar e verificar dados com RSA](#)
- [Derive keys using HMAC KDF](#)
- [Wrap and unwrap keys with AES using PKCS #5 padding](#)
- [Wrap and unwrap keys with AES using no padding](#)
- [Wrap and unwrap keys with AES using zero padding](#)
- [Wrap and unwrap keys with AES-GCM](#)
- [Encapsular e desencapsular chaves com RSA](#)

Configurações avançadas para a biblioteca PKCS #11 para AWS CloudHSM

O provedor AWS CloudHSM PKCS #11 inclui a seguinte configuração avançada, que não faz parte das configurações gerais que a maioria dos clientes utiliza. Essas configurações fornecem recursos adicionais.

- [Conectar-se a vários slots com o PKCS #11](#)
- [Repita a configuração do PKCS #11](#)

Configuração de vários slots com a biblioteca PKCS #11 para AWS CloudHSM

Um único slot na biblioteca PKCS #11 do Client SDK 5 representa uma única conexão com um cluster no AWS CloudHSM. Com o Client SDK 5, você pode configurar sua PKCS11 biblioteca para permitir que vários slots conectem usuários a vários clusters do CloudHSM a partir de um único aplicativo PKCS #11.

Use as instruções neste tópico para fazer com que seu aplicativo use a funcionalidade de vários slots para se conectar a vários clusters.

Tópicos

- [Pré-requisitos de vários slots para a biblioteca PKCS #11 para AWS CloudHSM](#)
- [Configure a biblioteca PKCS #11 para a funcionalidade de vários slots para AWS CloudHSM](#)
- [Adicione um cluster com funcionalidade de vários slots para AWS CloudHSM](#)
- [Remova um cluster com funcionalidade de vários slots para AWS CloudHSM](#)

Pré-requisitos de vários slots para a biblioteca PKCS #11 para AWS CloudHSM

Antes de configurar vários slots para a biblioteca PKCS #11 for AWS CloudHSM, preencha os seguintes pré-requisitos.

- Dois ou mais AWS CloudHSM clusters aos quais você gostaria de se conectar, junto com seus certificados de cluster.
- Uma instância do EC2 com grupos de segurança configurados corretamente para se conectar a todos os clusters acima. Para obter mais informações sobre como configurar um cluster e a instância cliente, consulte [Introdução ao AWS CloudHSM](#).
- Para configurar a funcionalidade de vários slots, você já deve ter baixado e instalado a biblioteca PKCS #11. Se ainda não tiver feito isso, consulte as instruções em [???](#).

Configure a biblioteca PKCS #11 para a funcionalidade de vários slots para AWS CloudHSM

Para configurar sua biblioteca PKCS #11 para a funcionalidade de vários slots AWS CloudHSM, siga estas etapas:

1. Identifique os clusters aos quais você deseja se conectar usando a funcionalidade de vários slots.
2. Adicione esses clusters à sua configuração PKCS #11 seguindo as instruções em [???](#)
3. Na próxima vez que seu aplicativo PKCS #11 for executado, ele terá a funcionalidade de vários slots.

Adicione um cluster com funcionalidade de vários slots para AWS CloudHSM

Ao [se conectar a vários slots com o PKCS #11](#) for AWS CloudHSM, use o `configure-pkcs11 add-cluster` comando para adicionar um cluster à sua configuração.

Sintaxe

```
configure-pkcs11 add-cluster [OPTIONS]
  --cluster-id <CLUSTER ID>
  [--region <REGION>]
  [--endpoint <ENDPOINT>]
  [--hsm-ca-cert <HSM CA CERTIFICATE FILE>]
  [--client-cert-hsm-tls-file <CLIENT CERTIFICATE FILE>]
  [--client-key-hsm-tls-file <CLIENT KEY FILE>]
  [-h, --help]
```

Exemplos

Adicione um cluster usando o parâmetro **cluster-id**

Example

Use o parâmetro `configure-pkcs11 add-cluster` junto com `cluster-id` para adicionar um cluster (com o ID `decluster-1234567`) para a sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 add-cluster --cluster-id <cluster-1234567>
```


Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" add-cluster --cluster-id <cluster-1234567>
```

Tip

Se o uso de `configure-pkcs11 add-cluster` com o parâmetro `cluster-id` não resultar na adição do cluster, consulte o exemplo a seguir para obter uma versão mais longa deste comando que também requer os parâmetros `--region` e `--endpoint` para identificar o cluster que está sendo adicionado. Se, por exemplo, a região do cluster for diferente daquela configurada como padrão da AWS CLI, você deverá usar o parâmetro `--region` para usar a região correta. Além disso, você pode especificar o endpoint da AWS CloudHSM API a ser usado na chamada, o que pode ser necessário para várias configurações de rede, como usar endpoints de interface VPC que não usam o nome de host DNS padrão para AWS CloudHSM.

Adicione um cluster usando os parâmetros **cluster-id**, **endpoint** e **region**

Example

Use `configure-pkcs11 add-cluster` junto com os parâmetros `cluster-id`, `endpoint` e `region` para adicionar um cluster (com o ID de `cluster-1234567`) à sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 add-cluster --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" add-cluster --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Para obter mais informações sobre os parâmetros `--endpoint`, `--cluster-id` e `--region`, consulte [the section called “Parâmetros”](#).

Parâmetros

`--ID do cluster` **<Cluster ID>**

Faz uma chamada `DescribeClusters` para encontrar todos os endereços IP da interface de rede elástica (ENI) do HSM no cluster associado ao ID do cluster. O sistema adiciona os endereços IP ENI aos arquivos de AWS CloudHSM configuração.

Note

Se você usar o `--cluster-id` parâmetro de uma instância do EC2 em uma VPC que não tem acesso à Internet pública, deverá criar uma interface VPC endpoint à qual se conectar. AWS CloudHSM Para obter mais informações sobre os endpoints da VPC, consulte [???](#).

Obrigatório: Sim

`--ponto final` **<Endpoint>**

Especifique o endpoint AWS CloudHSM da API usado para fazer a `DescribeClusters` chamada. Você deve definir essa opção em combinação com `--cluster-id`.

Obrigatório: não

`--hsm-ca-cert` **<HsmCA Certificate Filepath>**

Especifica o caminho de arquivo do certificado de CA do HSM.

Obrigatório: não

`--região` **<Region>**

Especifique a região do seu cluster. Você deve definir essa opção em combinação com `--cluster-id`.

Se você não fornecer o parâmetro `--region`, o sistema escolherá a região tentando ler as variáveis de ambiente `AWS_DEFAULT_REGION` ou `AWS_REGION`. Se essas variáveis não estiverem definidas, o sistema verificará a região associada ao seu perfil no seu arquivo de AWS Config (normalmente `~/.aws/config`), a menos que você tenha especificado um arquivo

diferente na variável de ambiente `AWS_CONFIG_FILE`. Se nenhuma das opções acima for definida, o sistema usará a região `us-east-1` como padrão.

Obrigatório: não

`-- client-cert-hsm-tls -arquivo <client certificate hsm tls path>`

Caminho para o certificado de cliente usado para autenticação mútua entre cliente TLS e HSM.

Use essa opção somente se você tiver registrado pelo menos uma âncora de confiança no HSM com a CLI do CloudHSM. Você deve definir essa opção em combinação com `--client-key-hsm-tls-file`.

Obrigatório: não

`-- client-key-hsm-tls -arquivo <client key hsm tls path>`

Caminho para a chave do cliente usado para autenticação mútua entre cliente TLS e HSM.

Use essa opção somente se você tiver registrado pelo menos uma âncora de confiança no HSM com a CLI do CloudHSM. Você deve definir essa opção em combinação com `--client-cert-hsm-tls-file`.

Obrigatório: não

Remova um cluster com funcionalidade de vários slots para AWS CloudHSM

Ao [se conectar a vários slots com o PKCS #11](#), use o comando `configure-pkcs11 remove-cluster` para remover um cluster dos slots PKCS #11 disponíveis.

Sintaxe

```
configure-pkcs11 remove-cluster [OPTIONS]
    --cluster-id <CLUSTER ID>
    [-h, --help]
```

Exemplos

Remover um cluster usando o parâmetro **cluster-id**

Example

Use o parâmetro `configure-pkcs11 remove-cluster` junto com `cluster-id` para remover um cluster (com o ID de `cluster-1234567`) da sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 remove-cluster --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" remove-cluster --cluster-id <cluster-1234567>
```

Para obter mais informações sobre o parâmetro `--cluster-id`, consulte [the section called “Parâmetros”](#).

Parâmetro

--ID do cluster **<Cluster ID>**

O ID do cluster a ser removido da configuração

Obrigatório: Sim

Tente novamente os comandos da biblioteca PKCS #11 para AWS CloudHSM

AWS CloudHSM O SDK do cliente 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações controladas pelo HSM do lado do cliente. Quando um HSM acelera as operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, o cliente SDKs tentará repetir as operações limitadas até 3 vezes enquanto recua exponencialmente. Essa estratégia de repetição automática pode ser configurada para um dos dois modos: desativado e padrão.

- desativado: o Client SDK não executará nenhuma estratégia de repetição para nenhuma operação com controle de utilização pelo HSM.
- padrão: esse é o modo padrão para o Client SDK 5.8.0 e versões posteriores. Nesse modo, o cliente SDKs repetirá automaticamente as operações limitadas recuando exponencialmente.

Para obter mais informações, consulte [Controle de utilização do HSM](#).

Definir os comandos de repetição para o modo desativado

Linux

Para definir comandos de repetição off para o Client SDK 5 no Linux

- Use os comandos a seguir para gerenciar as configurações do modo off:

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --default-retry-mode off
```

Windows

Para definir comandos de repetição off para o Client SDK 5 no Windows

- Use os comandos a seguir para gerenciar as configurações do modo off:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --default-retry-mode off
```

Armazenamento de certificados com a biblioteca PKCS #11

A biblioteca AWS CloudHSM PKCS #11 é compatível com o armazenamento de certificados de chave pública como “objetos públicos” (conforme definido no PKCS #11 2.40) em clusters hsm2m.medium. Esse atributo permite que sessões públicas e privadas do PKCS #11 criem, recuperem, modifiquem e excluam certificados de chave pública.

Para usar o armazenamento de certificados com a biblioteca PKCS #11, você precisa habilitá-lo na configuração do seu cliente. Depois de habilitado, você pode gerenciar objetos de certificado de suas aplicações PKCS #11. As operações que se aplicam tanto ao certificado quanto aos objetos de chave, como [C_FindObjects](#), retornarão resultados do armazenamento da chave e do certificado.

Tópicos

- [Ativar o armazenamento de certificados](#)
- [Operações da API de armazenamento de certificados](#)
- [Atributos de armazenamento de certificados](#)
- [Logs de auditoria de armazenamento de certificados](#)

Ativar o armazenamento de certificados

Você pode ativar o armazenamento de certificados em clusters hsm2m.medium usando a ferramenta de configuração da biblioteca PKCS #11. Esse atributo só está disponível no SDK versões 5.13 e posteriores. Para obter uma lista de operações que oferecem suporte ao tipo de objeto de certificado, consulte [Operações da API de armazenamento de certificados](#).

Para ativar o armazenamento de certificados, siga estas etapas para seu sistema operacional:

Linux

- Ativar o armazenamento de certificados

Execute este comando: .

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --enable-certificate-storage
```

Windows

- Ativar o armazenamento de certificados

Abra um prompt de comando e execute o seguinte comando:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --enable-certificate-storage
```

Operações da API de armazenamento de certificados

As seguintes operações do PKCS #11 oferecem suporte ao tipo de objeto certificado (CKO_CERTIFICATE):

Operações gerais de certificado

C_CreateObject

Cria um objeto de certificado.

C_DestroyObject

Exclui um objeto de certificado existente.

C_GetAttributeValue

Obtém o valor de um ou mais atributos de um objeto de certificado.

C_SetAttributeValue

Atualiza o valor de um ou mais atributos de um objeto de certificado.

Operações de pesquisa de objetos de certificado

C_FindObjectsInit

Inicia uma pesquisa por objetos de certificado.

C_FindObjects

Continua uma pesquisa por objetos de certificado.

C_FindObjectsFinal

Encerra uma pesquisa por objetos de certificado.

Atributos de armazenamento de certificados

A tabela a seguir lista os atributos do objeto de certificado compatíveis e seus valores:

Atributo	Valor padrão	Description
CKA_CLASS	Obrigatório	Deve ser CKO_CERTIFICATE .
CKA_TOKEN	Verdadeiro	Deve ser True.
CKA_MODIFIABLE	Verdadeiro	Deve ser True.
CKA_PRIVATE	Falso	Deve ser False.
CKA_LABEL	Vazio	Limite de 127 caracteres.
CKA_COPYABLE	Falso	Deve ser False.
CKA_DESTROYABLE	Verdadeiro	Deve ser True.

Atributo	Valor padrão	Description
CKA_CERTIFICATE_TYPE	Obrigatório	Deve ser CKC_X_509 .
CKA_TRUSTED	Falso	Deve ser False.
CKA_CERTIFICATE_CATEGORY	CK_CERTIFICATE_CATEGORY_UNSPECIFIED	Deve ser CK_CERTIFICATE_CATEGORY_UNSPECIFIED .
CKA_CHECK_VALUE	Derivado de CKA_VALUE	Definido automaticamente com base em CKA_VALUE .
CKA_START_DATE	Vazio	A data “não antes de” do certificado.
CKA_END_DATE	Vazio	A data “não depois de” do certificado.
CKA_PUBLIC_KEY_INFO	Vazio	O tamanho máximo é de 16 kilobytes.
CKA_SUBJECT	Obrigatório	O assunto do certificado.
CKA_ID	Vazio	O tamanho máximo é 128 bytes. A exclusividade não é imposta.
CKA_ISSUER	Vazio	O emissor do certificado.
CKA_SERIAL_NUMBER	Vazio	O número de série do certificado.
CKA_VALUE	Obrigatório	O tamanho máximo é de 32 kilobytes.

Logs de auditoria de armazenamento de certificados

O AWS CloudHSM grava registros de auditoria para operações de armazenamento de certificados que modificam dados em um stream de log separado do CloudWatch Amazon Events dentro do

grupo de registros do seu cluster CloudWatch . Esse fluxo de logs é nomeado para o cluster, não para um HSM específico dentro do cluster.

Para obter informações sobre como acessar os registros de auditoria CloudWatch, consulte [Trabalhando com Amazon CloudWatch Logs e AWS CloudHSM Audit Logs](#).

Campos de entrada de log

`object_handle`

O identificador exclusivo do objeto do certificado.

`op_code`

A operação executada ou tentada. Possíveis valores:

- `CreateObject`
- `DestroyObject`
- `SetAttributeValues`

`response`

OK se a operação foi bem-sucedida ou um dos seguintes tipos de erro:

- `DuplicateAttribute`
- `InvalidAttributeValue`
- `ObjectNotFound`
- `MaxObjectsReached`
- `InternalFailure`

`attributes`

Os atributos modificados, se houver.

`timestamp`

O tempo em que a operação ocorreu, em milissegundos, desde a época do Unix.

Exemplos de log de auditoria

CreateObject exemplo

```
{  
  "object_handle": 463180677312929947,
```

```
"op_code": "CreateObject",
"response": "OK",
"attributes": null,
"timestamp": 1725482483671
}
```

DestroyObject exemplo

```
{
  "object_handle": 463180677312929947,
  "op_code": "DestroyObject",
  "response": "OK",
  "attributes": null,
  "timestamp": 1725482484559
}
```

SetAttributeValues exemplo

```
{
  "object_handle": 463180678453346687,
  "op_code": "SetAttributeValues",
  "response": "OK",
  "attributes": [
    "Label"
  ],
  "timestamp": 1725482488004
}
```

Exemplo malsucedido CreateObject

```
{
  "object_handle": null,
  "op_code": "CreateObject",
  "response": "MaxObjectsReached",
  "attributes": null,
  "timestamp": 1726084937125
}
```

Mecanismo AWS CloudHSM dinâmico OpenSSL para cliente SDK 5

O AWS CloudHSM OpenSSL Dynamic Engine permite que você descarregue operações criptográficas em seu cluster CloudHSM por meio da API OpenSSL.

AWS CloudHSM fornece um mecanismo dinâmico OpenSSL, sobre o qual você pode ler em [ou](#). [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando Tomcat com JSSE](#) [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando NGINX ou Apache com OpenSSL](#) Para ver um exemplo de uso AWS CloudHSM com o OpenSSL, consulte [este blog de segurança da AWS](#). Para obter informações sobre o suporte da plataforma para SDKs, consulte [the section called “Plataformas compatíveis”](#). Para solução de problemas, consulte [Problemas conhecidos do OpenSSL Dynamic Engine para AWS CloudHSM](#).

Use as seções a seguir para instalar e configurar o mecanismo AWS CloudHSM dinâmico para OpenSSL usando o Client SDK 5.

Para obter mais informações sobre como usar Client SDK 3, consulte [Usando a versão anterior do SDK para trabalhar com AWS CloudHSM](#).

Tópicos

- [Instale o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5](#)
- [Tipos de chave compatíveis com o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5](#)
- [Mecanismos compatíveis com o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5](#)
- [Configurações avançadas para OpenSSL para AWS CloudHSM](#)

Instale o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5

Use as seções a seguir para instalar o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5.

Note

Para executar um único cluster HSM com o Client SDK 5, você deve primeiro gerenciar as configurações de durabilidade da chave do cliente configurando `disable_key_availability_check` como `True`. Para obter mais informações, consulte [Sincronização de chave](#) e a [ferramenta de configuração do Client SDK 5](#).

Instalar e configurar o mecanismo dinâmico do OpenSSL

1. Use os seguintes comandos para fazer download e instalar o mecanismo OpenSSL.

Amazon Linux 2023

Instale o OpenSSL Dynamic Engine para Amazon Linux 2023 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-dyn-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.amzn2023.x86_64.rpm
```

Instale o OpenSSL Dynamic Engine para Amazon Linux 2023 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-dyn-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Instale o OpenSSL Dynamic Engine para Amazon Linux 2 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-dyn-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el7.x86_64.rpm
```

Instale o OpenSSL Dynamic Engine para Amazon Linux 2 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-dyn-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

Instale o OpenSSL Dynamic Engine para RHEL 10 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-dyn-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el10.x86_64.rpm
```

Instale o OpenSSL Dynamic Engine para RHEL 10 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-dyn-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

Instale o OpenSSL Dynamic Engine para RHEL 9 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-dyn-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el9.x86_64.rpm
```

Instale o OpenSSL Dynamic Engine para RHEL 9 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-dyn-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

Instale o OpenSSL Dynamic Engine para RHEL 8 na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-dyn-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el8.x86_64.rpm
```

Instale o OpenSSL Dynamic Engine para RHEL 8 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-dyn-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el8.aarch64.rpm
```

Ubuntu 24.04 LTS

Instale o OpenSSL Dynamic Engine para Ubuntu 24.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-dyn_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u24.04_amd64.deb
```

Instale o OpenSSL Dynamic Engine para Ubuntu 24.04 LTS na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-dyn_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Instale o OpenSSL Dynamic Engine para Ubuntu 22.04 LTS na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-dyn_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u22.04_amd64.deb
```

Instale o OpenSSL Dynamic Engine para Ubuntu 22.04 LTS na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-dyn_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u22.04_arm64.deb
```

Você instalou a biblioteca compartilhada para o mecanismo dinâmico em `/opt/cloudhsm/lib/libcloudhsm_openssl_engine.so`.

2. Bootstrap do Client SDK 5. Para obter mais informações sobre ações de bootstrap, consulte [Bootstrap o Client SDK](#).
3. Defina uma variável de ambiente com as credenciais de um usuário de criptografia (CU). Para obter informações sobre a criação CUs, consulte [the section called “criar”](#).

```
$ export CLOUDHSM_PIN=<HSM user name>:<password>
```

Note

O Client SDK 5 introduz a variável de `CLOUDHSM_PIN` ambiente para armazenar as credenciais do CU. No Client SDK 3, você armazena as credenciais do CU na variável de `n3fips_password` ambiente. O Client SDK 5 é compatível com as duas variáveis de ambiente, mas recomendamos o uso de `CLOUDHSM_PIN`.

Ao definir variáveis de `CLOUDHSM_PIN` ambiente, você deve escapar de quaisquer caracteres especiais que possam ser interpretados pelo seu shell.

4. Conecte sua instalação do mecanismo dinâmico do OpenSSL ao cluster. Para obter mais informações, consulte [Conexão com um cluster](#).
5. Bootstrap o Client SDK 5. Para obter mais informações, consulte [the section called “Bootstrap o Client SDK”](#).

Verifique o mecanismo dinâmico do OpenSSL para Client SDK 5

Use o comando a seguir para verificar sua instalação do mecanismo dinâmico do OpenSSL.

```
$ openssl engine -t cloudhsm
```

A saída a seguir verifica sua configuração:

```
(cloudhsm) CloudHSM OpenSSL Engine  
[ available ]
```

Tipos de chave compatíveis com o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5

O AWS CloudHSM OpenSSL Dynamic Engine suporta os seguintes tipos de chave com o Client SDK 5.

Tipo de chave	Description
EC	ECDSA sign/verify para tipos de chaves P-256, P-384 e secp256k1. Para gerar chaves EC que sejam interoperáveis com o mecanismo OpenSSL, consulte Exportar uma chave assimétrica com a CloudHSM CLI .
RSA	Geração de chave de RSA para chaves de 2048, 3072 e 4096 bits. Assinar/verificar RSA. A verificação é descarregada para o software OpenSSL.

Mecanismos compatíveis com o OpenSSL Dynamic Engine AWS CloudHSM for Client SDK 5

O AWS CloudHSM OpenSSL Dynamic Engine suporta os seguintes mecanismos para as funções de assinatura e verificação com o Client SDK 5.

Funções de assinatura e verificação

Com o Client SDK 5, os dados são codificados localmente no software. Isso significa que não há limite em relação ao tamanho dos dados que podem ser codificados.

Tipos de assinatura RSA

- SHA1withRSA
- SHA224withRSA

- SHA256withRSA
- SHA384withRSA
- SHA512withRSA

Tipos de assinatura ECDSA

- SHA1withECDSA
- SHA224withECDSA
- SHA256withECDSA
- SHA384withECDSA
- SHA512withECDSA

Configurações avançadas para OpenSSL para AWS CloudHSM

O provedor AWS CloudHSM OpenSSL inclui a seguinte configuração avançada, que não faz parte das configurações gerais que a maioria dos clientes utiliza. Essas configurações fornecem recursos adicionais.

- [Repetir comandos para OpenSSL](#)

Tente novamente os comandos do OpenSSL para AWS CloudHSM

AWS CloudHSM O SDK do cliente 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações controladas pelo HSM do lado do cliente. Quando um HSM acelera as operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, o cliente SDKs tentará repetir as operações limitadas até 3 vezes enquanto recua exponencialmente. Essa estratégia de repetição automática pode ser configurada para um dos dois modos: desativado e padrão.

- desativado: o Client SDK não executará nenhuma estratégia de repetição para nenhuma operação com controle de utilização pelo HSM.
- padrão: esse é o modo padrão para o Client SDK 5.8.0 e versões posteriores. Nesse modo, o cliente SDKs repetirá automaticamente as operações limitadas recuando exponencialmente.

Para obter mais informações, consulte [Controle de utilização do HSM](#).

Definir os comandos de repetição para o modo desativado

Você pode usar os comandos a seguir para definir os comandos de repetição para o modo off:

```
$ sudo /opt/cloudhsm/bin/configure-dyn --default-retry-mode off
```

Provedor AWS CloudHSM de OpenSSL para cliente SDK 5

O AWS CloudHSM OpenSSL Provider permite que você descarregue operações criptográficas TLS em seu cluster CloudHSM por meio da API do OpenSSL Provider. A interface do provedor é a abordagem recomendada para novas implantações usando o OpenSSL 3.2 e versões posteriores.

Use as seções a seguir para instalar e configurar o AWS CloudHSM OpenSSL Provider usando o Client SDK 5.

Warning

Atualmente, a integração com o OpenSSL CLI não é suportada pelo OpenSSL Provider. AWS CloudHSM Consulte [the section called “Descarregue no Linux com o OpenSSL Provider”](#) as integrações compatíveis.

Plataformas compatíveis

O OpenSSL Provider requer o OpenSSL 3.2 ou posterior, disponível em +, Ubuntu 24.04+ e Amazon Linux EL9 2023+.

Verifique a compatibilidade: `openssl version`

Tópicos

- [Instale o OpenSSL Provider AWS CloudHSM for Client SDK 5](#)
- [Tipos de chave compatíveis com o OpenSSL Provider AWS CloudHSM for Client SDK 5](#)
- [Mecanismos suportados pelo OpenSSL Provider](#)
- [Configuração avançada do OpenSSL Provider](#)

Instale o OpenSSL Provider AWS CloudHSM for Client SDK 5

Use as seções a seguir para instalar o provedor OpenSSL para o SDK 5 do cliente AWS CloudHSM.

Note

Para executar um único cluster HSM com o Client SDK 5, você deve primeiro gerenciar as configurações de durabilidade da chave do cliente configurando `disable_key_availability_check` como `True`. Para obter mais informações, consulte [Sincronização de chave](#) e a [ferramenta de configuração do Client SDK 5](#).

Requisitos

O provedor OpenSSL exige os tipos de cluster `hsm2m.medium` e o SDK mínimo do CloudHSM Client versão 5.17.0 ou posterior.

Instale o provedor OpenSSL

Para instalar o OpenSSL Provider

1. Use os comandos a seguir para baixar e instalar o OpenSSL Provider.

Amazon Linux 2023

Instale o provedor OpenSSL para Amazon Linux 2023 na arquitetura `x86_64`:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-openssl-provider-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.amzn2023.x86_64.rpm
```

Instale o provedor OpenSSL para Amazon Linux 2023 na arquitetura: `ARM64`

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-openssl-provider-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.amzn2023.aarch64.rpm
```

RHEL 9 (9.2+)

Instale o provedor OpenSSL para RHEL 9 na arquitetura `x86_64`:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-openssl-provider-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el9.x86_64.rpm
```

Instale o provedor OpenSSL para RHEL 9 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-openssl-provider-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el9.aarch64.rpm
```

RHEL 10 (10.0+)

Instale o provedor OpenSSL para RHEL 10 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-openssl-provider-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el10.x86_64.rpm
```

Instale o provedor OpenSSL para RHEL 10 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-openssl-provider-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el10.aarch64.rpm
```

Ubuntu 24.04

Instale o provedor OpenSSL para Ubuntu 24.04 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-openssl-provider_latest_u24.04_amd64.deb
```

```
$ sudo dpkg -i ./cloudhsm-openssl-provider_latest_u24.04_amd64.deb
```

Instale o provedor OpenSSL para Ubuntu 24.04 na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-openssl-provider_latest_u24.04_arm64.deb
```

```
$ sudo dpkg -i ./cloudhsm-openssl-provider_latest_u24.04_arm64.deb
```

Você instalou a biblioteca compartilhada para o OpenSSL Provider em. /opt/cloudhsm/lib/licloudhsm_openssl_provider.so

2. Bootstrap do Client SDK 5. Para obter mais informações sobre ações de bootstrap, consulte [Bootstrap o Client SDK](#).
3. Defina a variável de CLOUDHSM_PIN ambiente com suas credenciais de usuário criptográfico (CU):

```
$ export CLOUDHSM_PIN=<username>:<password>
```

4. Conecte sua instalação do OpenSSL Provider ao cluster. Para obter mais informações, consulte [Conexão com um cluster](#).

Verificar a instalação

Verifique se o OpenSSL Provider está instalado corretamente:

```
$ CLOUDHSM_PIN=<username>:<password> openssl list -providers -provider cloudhsm
```

Você deve ver uma saída semelhante a:

```
Providers:
  cloudhsm
    name: AWS CloudHSM OpenSSL Provider
    version: 5.17.0
    status: active
  default
    name: OpenSSL Default Provider
    version: 3.2.2
```

```
status: active
```

Tipos de chave compatíveis com o OpenSSL Provider AWS CloudHSM for Client SDK 5

O AWS CloudHSM OpenSSL Provider oferece suporte aos seguintes tipos de chave com o Client SDK 5.

Tipo de chave	Description
RSA	Operações de criptografia RSA sign/verify e assimétrica. A verificação é descarregada para o software OpenSSL. Para gerar chaves RSA que sejam interoperáveis com o provedor OpenSSL, consulte. Exportar uma chave assimétrica com a CloudHSM CLI
EC	ECDSA sign/verify para curvas P-256, P-384 e P-521. A verificação é descarregada para o software OpenSSL. Para gerar chaves EC que sejam interoperáveis com o provedor OpenSSL, consulte. Exportar uma chave assimétrica com a CloudHSM CLI

Mecanismos suportados pelo OpenSSL Provider

O AWS CloudHSM OpenSSL Provider SDK oferece suporte a um conjunto abrangente de mecanismos criptográficos para várias operações, incluindo assinaturas digitais, criptografia assimétrica, criptografia simétrica, troca de chaves e muito mais.

Tipos de assinatura RSA

O OpenSSL Provider oferece suporte a assinaturas digitais RSA com vários algoritmos de hash e esquemas de preenchimento:

SHA1withRSA

Assinaturas RSA com algoritmo de hash SHA-1

- Preenchimento PKCS #1 v1.5
- Preenchimento PSS (Esquema de Assinatura Probabilística)

SHA224withRSA

Assinaturas RSA com algoritmo de hash SHA-224

- Preenchimento PKCS #1 v1.5
- Estofamento PSS

SHA256withRSA

Assinaturas RSA com algoritmo de hash SHA-256

- Preenchimento PKCS #1 v1.5
- Estofamento PSS

SHA384withRSA

Assinaturas RSA com algoritmo de hash SHA-384

- Preenchimento PKCS #1 v1.5
- Estofamento PSS

SHA512withRSA

Assinaturas RSA com algoritmo de hash SHA-512

- Preenchimento PKCS #1 v1.5
- Estofamento PSS

Tipos de assinatura ECDSA

O OpenSSL Provider oferece suporte a assinaturas digitais ECDSA com vários algoritmos de hash:

SHA1withECDSA

Assinaturas ECDSA com algoritmo de hash SHA-1

SHA224withECDSA

Assinaturas ECDSA com algoritmo de hash SHA-224

SHA256withECDSA

Assinaturas ECDSA com algoritmo de hash SHA-256

SHA384withECDSA

Assinaturas ECDSA com algoritmo de hash SHA-384

SHA512withECDSA

Assinaturas ECDSA com algoritmo de hash SHA-512

Configuração avançada do OpenSSL Provider

O SDK do provedor de OpenSSL do AWS CloudHSM fornece opções avançadas de configuração para personalizar seu comportamento para diferentes casos de uso e ambientes. Use a ferramenta de configuração para definir essas opções.

- [Tente novamente a configuração do OpenSSL Provider](#)

Tente novamente os comandos do OpenSSL Provider para AWS CloudHSM

AWS CloudHSM O SDK do cliente 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações controladas pelo HSM do lado do cliente. Quando um HSM acelera as operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, o cliente SDKs tentará repetir as operações limitadas até 3 vezes enquanto recua exponencialmente. Essa estratégia de repetição automática pode ser configurada para um dos dois modos: desativado e padrão.

- desativado: o Client SDK não executará nenhuma estratégia de repetição para nenhuma operação com controle de utilização pelo HSM.
- padrão: esse é o modo padrão para o Client SDK 5.8.0 e versões posteriores. Nesse modo, o cliente SDKs repetirá automaticamente as operações limitadas recuando exponencialmente.

Para obter mais informações, consulte [Controle de utilização do HSM](#).

Definir os comandos de repetição para o modo desativado

Linux

Para definir comandos de repetição off para o Client SDK 5 no Linux

- Use os comandos a seguir para gerenciar as configurações do modo off:


```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --default-retry-mode off
```

Provedor de armazenamento de chaves (KSP) para o AWS CloudHSM Client SDK 5

O Provedor de armazenamento de chaves (KSP) é uma API de criptografia específica para o sistema operacional Microsoft Windows. O Provedor de armazenamento de chaves (KSP) permite que os desenvolvedores usem técnicas de criptografia para proteger aplicações baseadas em Windows.

Para obter informações sobre bootstrapping, consulte [Conectar-se ao cluster](#).

Para obter mais informações sobre como usar Client SDK 3, consulte [Usando a versão anterior do SDK para trabalhar com AWS CloudHSM](#).

Tópicos

- [Instale o provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#)
- [Autentique-se no provedor de armazenamento de chaves \(KSP\) do AWS CloudHSM Client SDK 5](#)
- [Tipos de chaves compatíveis com o Key Storage Provider \(KSP\) para AWS CloudHSM Client SDK 5](#)
- [Operações de API suportadas Provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#)
- [Configurações avançadas do KSP para AWS CloudHSM](#)

Instale o provedor de armazenamento de chaves (KSP) para o AWS CloudHSM Client SDK 5

Use as seções a seguir para instalar o provedor de armazenamento de chaves (KSP) para o AWS CloudHSM Client SDK 5.

Note

Para executar um único cluster HSM com o Client SDK 5, você deve primeiro gerenciar as configurações de durabilidade da chave do cliente configurando

`disable_key_availability_check` como `True`. Para obter mais informações, consulte [Sincronização de chave](#) e a [ferramenta de configuração do Client SDK 5](#).

Para instalar e configurar o Provedor de armazenamento de chaves (KSP)

1. Instale o Key Storage Provider (KSP) para Windows Server na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMKSP-latest.msi -Outfile C:\AWSCloudHSMKSP-latest.msi
```

```
PS C:\> Start-Process msixexec.exe -ArgumentList '/i C:\AWSCloudHSMKSP-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

2. Use a ferramenta de configuração para especificar um local para o certificado de emissão. Para instruções, consulte [Especifique o local do certificado de emissão..](#)
3. Para se conectar e usar o cluster [Bootstrap o Client SDK](#).
4. Você pode encontrar os arquivos do Provedor de armazenamento de chaves (KSP) nos seguintes locais:

- Binários do Windows:

```
C:\Program Files\Amazon\CloudHSM
```

Scripts de configuração e arquivos de log do Windows:

```
C:\ProgramData\Amazon\CloudHSM
```

Autentique-se no provedor de armazenamento de chaves (KSP) do AWS CloudHSM Client SDK 5

Antes de usar o provedor de armazenamento de chaves (KSP) para o AWS CloudHSM Client SDK 5, você deve definir as credenciais de login para o HSM em seu sistema. Você tem duas opções:

- Gerenciador de credenciais do Windows (recomendado para maior segurança)
- Variáveis de ambiente do sistema (configuração mais simples)

Gerenciador de credenciais do Windows

Você pode configurar as credenciais usando o utilitário `set_cloudhsm_credentials` ou a interface do Gerenciador de credenciais do Windows.

- Usar o utilitário **`set_cloudhsm_credentials`**:

O Windows Installer inclui o utilitário `set_cloudhsm_credentials`. É possível usar esse utilitário para passar convenientemente as credenciais de login do HSM para o Gerenciador de credenciais do Windows. Se desejar compilar esse utilitário da origem, use o código Python incluído no instalador.

1. Acesse `C:\Program Files\Amazon\CloudHSM\tools\`.
2. Execute este comando: .

```
set_cloudhsm_credentials.exe --username <CU USER> --password <CU PASSWORD>
```

- Usar a interface do Gerenciador de credenciais:

1. Abra o Gerenciador de credenciais:
 - Insira `credential manager` na caixa de pesquisa da barra de tarefas
 - Selecione Gerenciador de credenciais
2. Selecione Windows Credentials (Credenciais do Windows) para gerenciar as credenciais do Windows.
3. Selecione Adicionar uma credencial genérica
4. Insira os detalhes a seguir:
 - Endereço de internet ou de rede: `CLOUDHSM_PIN`.
 - Nome de usuário: <CU USER>.
 - Senha: <CU PASSWORD>.
5. Escolha OK

Variáveis de ambiente do sistema

Você pode definir as variáveis de ambiente do sistema que identificam um HSM e um [usuário de criptografia](#) (CU).

 Warning

A configuração de credenciais por meio de variáveis de ambiente do sistema armazena sua senha em texto simples no sistema. Para maior segurança, use o Gerenciador de credenciais do Windows em vez disso.

Você pode definir as variáveis de ambiente usando:

- O [setx](#).
- O painel de controle Propriedades do Sistema (guia Avançado) do Windows.
- definir métodos [Programáticos](#) de variáveis de ambiente do sistema permanentes.

Para definir a variável de ambiente do sistema:

CLOUDHSM_PIN=<CU USERNAME>:<CU PASSWORD>

Identifica um [crypto user \(usuário de criptografia\)](#) (CU) no HSM e fornece todas as informações de login necessárias. O aplicativo é autenticado e executado como esse CU. O aplicativo tem as permissões desse CU e pode visualizar e gerenciar apenas as chaves que o CU possui e compartilha. Para criar um usuário de criptografia, use o comando [user create](#) na CloudHSM CLI. Para encontrar os existentes CUs, use o comando [user list](#) na CLI do CloudHSM.

Por exemplo:

```
setx /m CLOUDHSM_PIN test_user:password123
```

 Note

Ao definir as variáveis de ambiente CLOUDHSM_PIN, você deve escapar de quaisquer caracteres especiais que possam ser interpretados pelo seu shell.

Tipos de chaves compatíveis com o Key Storage Provider (KSP) para AWS CloudHSM Client SDK 5

O AWS CloudHSM Key Storage Provider (KSP) oferece suporte aos seguintes tipos de chaves com o Client SDK 5.

Tipo de chave	Description
EC	Gere chaves com as curvas secp256r1 (P-256), secp384r1 (P-384) e secp521r1 (P-521).
RSA	Gere chaves RSA de 2048, 3072 e 4096 bits.

Operações de API suportadas Provedor de armazenamento de chaves (KSP) para o AWS CloudHSM Client SDK 5

Os parâmetros no KSP são definidos pelo Microsoft KSP. Consulte a [documentação da Microsoft](#) para obter mais informações.

O Key Storage Provider (KSP) oferece suporte às seguintes operações da API KSP para o AWS CloudHSM Client SDK 5.

- [NCryptOpenStorageProvider](#)
- [NCryptOpenKey](#)
- [NCryptCreatePersistedKey](#)
- [NCryptGetProperty](#)
- [NCryptSetProperty](#)
- [NCryptFinalizeKey](#)
- [NCryptDeleteKey](#)
- [NCryptFreeObject](#)
- [NCryptFreeBuffer](#)
- [NCryptIsAlgSupported](#)
- [NCryptEnumAlgorithms](#)

- [NCryptEnumKeys](#)
- [NCryptExportKey](#)
- [NCryptSignHash](#)
- [NCryptVerifySignature](#)

NCryptOpenStorageProvider função com Key Storage Provider (KSP)

A função `NCryptOpenStorageProvider` carrega e inicializa o Provedor de armazenamento de chaves (KSP).

Parâmetros

`phProvider` [out]

Um ponteiro para uma variável `NCRYPT_PROV_HANDLE` que armazena o identificador do provedor.

`pszProviderName` [in]

Um ponteiro para uma string Unicode terminada em nulo identificando o provedor de armazenamento de chaves. O Provedor de armazenamento de chaves (KSP) do AWS CloudHSM oferece suporte aos seguintes valores:

Valor	Significado
L"CloudHSM Key Storage Provider"	Identifica o nome do provedor do Client SDK 5. Recomendamos usar esse nome por padrão.
L"Cavium Key Storage Provider"	Identifica o nome do provedor do Client SDK 3. Com suporte para compatibilidade com versões anteriores.



Note

Os valores são literais de string largos, conforme indicado pelo L antes do literal.

dwFlags [in]

Sinalizadores que modificam o comportamento da função. Nenhum sinalizador está definido para essa função.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.

NCryptOpenKey com o provedor de armazenamento de chaves (KSP)

A função NCryptOpenKey abre uma chave que existe no Provedor de Armazenamento de Chaves (KSP).

Parâmetros

hProvider [in]

O identificador do KSP que contém a chave. Use [NCryptOpenStorageProvider](#) para obter o identificador.

phKey [out]

Um ponteiro para uma variável NCryptKeyHandle que armazena o identificador da chave.

pszKeyName [in]

Um ponteiro para uma string Unicode terminada em nulo contendo o nome da chave.

dwLegacyKeySpec [in, unused]

AWS CloudHSM O Key Storage Provider (KSP) não usa esse parâmetro.

dwFlags [in]

Sinalizadores que modificam o comportamento da função. Nenhum sinalizador está definido para essa função.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_INVALID_HANDLE	O identificador em hProvider não é válido.
NTE_BAD_KEYSET	O nome da chave fornecido não retornou um resultado exclusivo.

NCryptCreatePersistedKey com o provedor de armazenamento de chaves (KSP)

A função NCryptCreatePersistedKey cria uma nova chave e a armazena no Provedor de armazenamento de chaves (KSP). Você pode usar a função [NCryptSetProperty](#) para definir suas propriedades após a criação. Você deve chamar [NCryptFinalizeKey](#) antes de poder usar a chave.

Parâmetros

hProvider [in]

O identificador do provedor de armazenamento de chaves em que você criará a chave. Use [NCryptOpenStorageProvider](#) para obter esse identificador.

phKey [out]

O endereço de uma variável NCryptKeyHandle que armazena o identificador da chave.

pszAlgId [in]

Um ponteiro para uma string Unicode terminada em nulo que especifica o identificador do algoritmo criptográfico para criar a chave.

AWS CloudHSM O Key Storage Provider (KSP) suporta os seguintes algoritmos:

Constante/Valor	Description
BCRYPT_RSA_ALGORITHM "RSA"	O algoritmo de chave pública do RSA.
BCRYPT_ECDSA_P256_ALGORITHM "ECDSA_P256"	O algoritmo de assinatura digital de curva elíptica principal de 256 bits (FIPS 186-2).
BCRYPT_ECDSA_P384_ALGORITHM "ECDSA_P384"	O algoritmo de assinatura digital de curva elíptica principal de 384 bits (FIPS 186-2).
BCRYPT_ECDSA_P521_ALGORITHM "ECDSA_P521"	O algoritmo de assinatura digital de curva elíptica principal de 521 bits (FIPS 186-2).

pszKeyName [in, optional]

Um ponteiro para uma string Unicode terminada em nulo que contém o nome da chave. Se esse parâmetro for NULL, essa função criará uma chave efêmera que não será persistida.

dwLegacyKeySpec [in, unused]

AWS CloudHSM O Key Storage Provider (KSP) não usa esse parâmetro.

dwFlags [in]

Sinalizadores para modificar o comportamento da função. Use nenhum ou mais dos seguintes valores:

Valor	Significado
NCRYPT_MACHINE_KEY_FLAG	Esse sinalizador não tem efeito.

Valor	Significado
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.
NCRYPT_OVERWRITE_KEY_FLAG	A especificação desse sinalizador sobrescreve qualquer chave existente com o mesmo nome no HSM. Sem esse sinalizador, a função retorna NTE_EXISTS.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A função foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.
NTE_NOT_SUPPORTED	O parâmetro pszAlgId contém um valor não suportado.
NTE_EXISTS	Já existe uma chave com o nome especificado e a operação não usou NCRYPT_OVERWRITE_KEY_FLAG.

NCryptGetProperty com o provedor de armazenamento de chaves (KSP)

A função NCryptGetProperty recupera os valores das propriedades de um objeto de armazenamento de chaves.

Parâmetros

hobject [in]

O identificador do objeto cuja propriedade você deseja recuperar. Você pode usar:

- Um identificador de provedor (NCRYPT_PROV_HANDLE)
- Um identificador de chave (NCRYPT_KEY_HANDLE)

pszProperty [in]

Um ponteiro para uma string Unicode terminada em nulo contendo o nome da propriedade a ser recuperada.

Ao usar NCRYPT_PROV_HANDLE, o AWS CloudHSM Key Storage Provider (KSP) oferece suporte aos seguintes identificadores KSP:

Identificador/Valor	Description
NCRYPT_IMPL_TYPE_PROPERTY L"Impl Type"	Um DWORD contendo sinalizadores que definem detalhes de implementação do provedor
NCRYPT_MAX_NAME_LENGTH_PROPERTY L"Max Name Length"	Um DWORD contendo o tamanho máximo (em caracteres) de um nome de chave persistente.
NCRYPT_NAME_PROPERTY L"Name"	Um ponteiro para uma string Unicode terminada em nulo contendo o nome do KSP.
NCRYPT_VERSION_PROPERTY L"Version"	Um DWORD contendo a versão do provedor (palavra em maiúsculas: versão principal; palavra em minúsculas: versão secundária).
NCRYPT_USE_CONTEXT_PROPERTY L"Use Context"	Um ponteiro para uma string Unicode terminada em nulo descrevendo o contexto da operação.
NCRYPT_SECURITY_DESCR_SUPPORT_PROPERTY	Indica se o provedor oferece suporte a descritores de segurança para chaves.

Identificador/Valor	Description
L"Security Descr Support"	

Ao usar `NCRYPT_KEY_HANDLE`, o AWS CloudHSM Key Storage Provider (KSP) oferece suporte aos seguintes identificadores KSP:

Identificador/Valor	Description
NCRYPT_ALGORITHM_PROPERTY L"Algorithm Name"	Um ponteiro para uma string Unicode terminada em nulo contendo o nome do algoritmo da chave.
NCRYPT_BLOCK_LENGTH_PROPERTY L"Block Length"	Um DWORD contendo o tamanho do bloco de criptografia em bytes.
NCRYPT_EXPORT_POLICY_PROPERTY L"Export Policy"	Um DWORD contendo sinalizadores que especificam a política de exportação da chave persistente.
NCRYPT_KEY_USAGE_PROPERTY L"Key Usage"	Um DWORD contendo sinalizadores que definem detalhes de uso da chave.
NCRYPT_KEY_TYPE_PROPERTY L"Key Type"	Um DWORD contendo sinalizadores que definem o tipo de chave.
NCRYPT_LENGTH_PROPERTY L"Length"	Um DWORD contendo o comprimento da chave em bits.
NCRYPT_LENGTHS_PROPERTY L"Lengths"	Um ponteiro para uma estrutura <code>NCRYPT_SUPPORTED_LENGTHS</code> contendo tamanhos de chave compatíveis.

Identificador/Valor	Description
NCRYPT_NAME_PROPERTY L"Name"	Um ponteiro para uma string Unicode terminada em nulo contendo o nome da chave.
NCRYPT_SECURITY_DESCR_PROPERTY L"Security Descr"	Um ponteiro para uma estrutura SECURITY_DESCRIPTOR contendo as principais informações de controle de acesso.
NCRYPT_ALGORITHM_GROUP_PROPERTY L"Algorithm Group"	Uma string Unicode terminada em nulo contendo o nome do grupo de algoritmos do objeto.
NCRYPT_UNIQUE_NAME_PROPERTY L"Unique Name"	Um ponteiro para uma string Unicode terminada em nulo contendo o nome exclusivo da chave.

 Note

Os valores são literais de string largos, conforme indicado pelo L antes do literal.

pbOutput [out]

O endereço de um buffer para armazenar o valor da propriedade. Especifique o tamanho do buffer usando cbOutput.

Para determinar o tamanho do buffer necessário, defina esse parâmetro como NULL. A função armazena o tamanho necessário (em bytes) no local apontado por pcbResult.

cbOutput [in]

O tamanho em bytes do buffer pbOutput.

pcbResult [out]

Um ponteiro para uma variável DWORD que armazena o número de bytes copiados para o buffer pbOutput.

Se pbOutput for NULL, isso armazenará o tamanho necessário (em bytes).

dwFlags [in]

Sinalizadores para modificar o comportamento da função. Você pode usar zero ou:

Valor	Significado
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.

Quando pszProperty for NCRYPT_SECURITY_DESCR_PROPERTY, use um ou uma combinação de:

Valor	Significado
OWNER_SECURITY_INFORMATION	Esse sinalizador não tem efeito.
GROUP_SECURITY_INFORMATION	Esse sinalizador não tem efeito.
DACL_SECURITY_INFORMATION	Esse sinalizador não tem efeito.
LABEL_SECURITY_INFORMATION	Esse sinalizador não tem efeito.
SACL_SECURITY_INFORMATION	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.

Código de retorno	Description
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.
NTE_NOT_SUPPORTED	O parâmetro pszAlgId contém um valor que não é suportado.
NTE_INVALID_HANDLE	O identificador em hObject não é válido.
NTE_BUFFER_TOO_SMALL	O parâmetro cbOutput é muito pequeno para valores de retorno.

NCryptSetProperty com o provedor de armazenamento de chaves (KSP)

A função NCryptSetProperty define os valores das propriedades para um objeto de armazenamento de chaves.

Parâmetros

hObject [in]

O identificador do objeto cuja propriedade você deseja definir. Você pode usar:

- Um identificador de provedor (NCRYPT_PROV_HANDLE)
- Um identificador de chave (NCRYPT_KEY_HANDLE)

pszProperty [in]

Um ponteiro para uma string Unicode terminada em nulo contendo o nome da propriedade a ser recuperada.

Ao usar NCRYPT_PROV_HANDLE, o AWS CloudHSM Key Storage Provider (KSP) oferece suporte aos seguintes identificadores KSP:

Identificador/Valor	Description
NCRYPT_USE_CONTEXT_PROPERTY L"Use Context"	Um ponteiro para uma string Unicode terminada em nulo descrevendo o contexto da operação.

Ao usar `NCRYPT_KEY_HANDLE`, o AWS CloudHSM Key Storage Provider (KSP) oferece suporte aos seguintes identificadores KSP:

Identificador/Valor	Description
<code>NCRYPT_KEY_USAGE_PROPERTY</code> <code>L"Key Usage"</code>	Um DWORD contendo um conjunto de sinalizadores que definem os detalhes de uso da chave. Essa propriedade só se aplica às chaves. Isso pode conter nenhuma ou uma combinação de um ou mais dos valores a seguir. <code>NCRYPT_ALLOW_DECRYPT_FLAG</code> (0x00000001) <code>NCRYPT_ALLOW_SIGNING_FLAG</code> (0x00000002)
<code>NCRYPT_LENGTH_PROPERTY</code> <code>L"Length"</code>	Um DWORD contendo o comprimento da chave em bits.
<code>NCRYPT_EXPORT_POLICY_PROPERTY</code> <code>L"Export Policy"</code>	Um DWORD contendo sinalizadores que especificam a política de exportação da chave persistente. Isso pode conter nenhuma ou uma combinação de um ou mais dos valores a seguir. <code>NCRYPT_ALLOW_EXPORT_FLAG</code> (0x00000001)

 Note

Os valores são literais de string largos, conforme indicado pelo L antes do literal.

pbInput [in]

O endereço de um buffer que contém o novo valor da propriedade. cbInput contém o tamanho do buffer.

cbInput [in]

O tamanho em bytes do buffer pbInput.

dwFlags [in]

Sinalizadores que modificam o comportamento da função. Nenhum sinalizador está definido para essa função.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.
NTE_NOT_SUPPORTED	O parâmetro pszProperty contém um valor que não é suportado.
NTE_INVALID_HANDLE	O identificador em hObject não é válido.
NTE_BAD_DATA	Os dados apontados por pbInput e cbInput não são válidos.

NCryptFinalizeKey com o provedor de armazenamento de chaves (KSP)

A função `NCryptFinalizeKey` completa uma chave do KSP. Você deve chamar essa função antes de poder usar a chave.

Parâmetros

`hKey` [in]

O identificador da chave para completar. Obtenha esse identificador chamando a [NCryptCreatePersistedKey](#) função.

`dwFlags` [in]

Sinalizadores para modificar o comportamento da função. Você pode usar nenhum ou um destes valores:

Valor	Significado
<code>NCRYPT_SILENT_FLAG</code>	Esse sinalizador não tem efeito.
<code>NCRYPT_NO_KEY_VALIDATION</code>	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
<code>ERROR_SUCCESS</code>	A operação foi concluída com êxito.
<code>NTE_FAIL</code>	A operação não foi concluída.
<code>NTE_INVALID_HANDLE</code>	O identificador em <code>hKey</code> não é válido.
<code>NTE_NOT_SUPPORTED</code>	O parâmetro <code>dwFlags</code> contém um valor que não é suportado.

Código de retorno	Description
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.

NCryptDeleteKey com o provedor de armazenamento de chaves (KSP)

A função NCryptDeleteKey exclui uma chave KSP do Provedor de armazenamento de chaves (KSP).

Parâmetros

hKey [in]

O identificador da chave a ser excluída.

dwFlags [in]

Sinalizadores para modificar o comportamento da função. É possível usar zero ou mais dos seguintes valores:

Valor	Significado
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A função foi bem-sucedida.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.

Código de retorno	Description
NTE_FAIL	A operação não foi concluída.
NTE_INVALID_HANDLE	O identificador em hKey não é válido.
NTE_INTERNAL_ERROR	Ocorreu um erro interno ao excluir a chave.

NCryptFreeObject com o provedor de armazenamento de chaves (KSP)

A função `NCryptFreeObject` libera o provedor ou o identificador de chave do Provedor de Armazenamento de Chaves (KSP).

Parâmetros

`hObject [in]`

O identificador do objeto a ser liberado. Você pode usar:

- Um identificador de provedor (`NCRYPT_PROV_HANDLE`)
- Um identificador de chave (`NCRYPT_KEY_HANDLE`)

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_HANDLE	O identificador em <code>hObject</code> não é válido.

NCryptFreeBuffer com o provedor de armazenamento de chaves (KSP)

A função `NCryptFreeBuffer` libera um bloco de memória alocado pelo Provedor de armazenamento de chaves (KSP).

Parâmetros

pvInput [in]

O endereço da memória a ser liberada.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_FAIL	A operação não foi concluída.

NCryptIsAlgSupported com o provedor de armazenamento de chaves (KSP)

NCryptIsAlgSupported função determina se o Key Storage Provider (KSP) suporta um algoritmo criptográfico específico.

Parâmetros

hProvider [in]

O identificador do provedor de armazenamento de chaves. Use [NCryptOpenStorageProvider](#) para obter o identificador.

pszAlgId [in]

Um ponteiro para uma string Unicode terminada em nulo que contém o identificador do algoritmo criptográfico para criar a chave. O Provedor de armazenamento de chaves (KSP) do AWS CloudHSM oferece suporte aos seguintes algoritmos:

Constante/Valor	Description
BCRYPT_RSA_ALGORITHM	O algoritmo de chave pública do RSA.
"RSA"	

Constante/Valor	Description
BCRYPT_ECDSA_P256_ALGORITHM "ECDSA_P256"	O algoritmo de assinatura digital de curva elíptica principal de 256 bits (FIPS 186-2).
BCRYPT_ECDSA_P384_ALGORITHM "ECDSA_P384"	O algoritmo de assinatura digital de curva elíptica principal de 384 bits (FIPS 186-2).
BCRYPT_ECDSA_P521_ALGORITHM "ECDSA_P521"	O algoritmo de assinatura digital de curva elíptica principal de 521 bits (FIPS 186-2).

dwFlags [in]

Sinalizadores que modificam o comportamento da função. Esse valor pode ser zero ou os seguintes:

Valor	Significado
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.

Código de retorno	Description
NTE_NOT_SUPPORTED	O parâmetro <code>pszAlgId</code> contém um valor não suportado.
NTE_INVALID_HANDLE	O identificador em <code>hProvider</code> não é válido.

NCryptEnumAlgorithms com o provedor de armazenamento de chaves (KSP)

A função `NCryptEnumAlgorithms` recupera os nomes dos algoritmos que o Provedor de armazenamento de chaves (KSP) suporta.

Parâmetros

`hProvider` [in]

O identificador do provedor de armazenamento de chaves para o qual enumerar os algoritmos. Use a função [NCryptOpenStorageProvider](#) para obter esse identificador.

`dwAlgOperations` [in]

Um conjunto de valores que especificam quais classes de algoritmo devem ser enumeradas. Use zero para enumerar todos os algoritmos ou combinar um ou mais destes valores:

Valor	Significado
NCRYPT_ASYMMETRIC_ENCRYPTION_OPERATION 0x00000004	Liste os algoritmos de criptografia assimétrica.
NCRYPT_SIGNATURE_OPERATION 0x00000010	Liste os algoritmos de assinatura digital.

`pdwAlgCount` [out]

O endereço de um `DWORD` que armazena o número de elementos na matriz `ppAlgList`.

ppAlgList [out]

O endereço de um ponteiro de estrutura NCryptAlgorithmName que armazena uma matriz de nomes de algoritmos registrados. O parâmetro pdwAlgCount indica o número de elementos nessa matriz.

dwFlags [in]

Sinalizadores para modificar o comportamento da função. Use zero ou o seguinte valor:

Valor	Significado
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.
NTE_NOT_SUPPORTED	O parâmetro dwAlgOperations contém um valor não suportado.

NCryptEnumKeys com o provedor de armazenamento de chaves (KSP)

NCryptEnumKeys função lista as chaves armazenadas no Provedor de Armazenamento de Chaves (KSP).

Parâmetros

hProvider [in]

O identificador do provedor de armazenamento de chaves. Use [NCryptOpenStorageProvider](#) para obter esse identificador.

pszScope [in, unused]

Defina este parâmetro como NULL.

ppKeyName [out]

Um endereço de ponteiro para uma estrutura NCryptKeyName que armazena o nome da chave. Para liberar essa memória após o uso, chame [NCryptFreeBuffer](#).

ppEnumState [in, out]

Um endereço de ponteiro VOID que rastreia o progresso da enumeração. O provedor de armazenamento de chaves usa essas informações internamente para gerenciar a sequência de enumeração. Para iniciar uma nova enumeração desde o início, defina esse ponteiro como NULL.

Para liberar essa memória após concluir a enumeração, passe esse ponteiro para [NCryptFreeBuffer](#).

dwFlags [in]

Sinalizadores para modificar o comportamento da função. A função não tem sinalizadores.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_INVALID_HANDLE	O identificador em hProvider não é válido.

Código de retorno	Description
NTE_NO_MORE_ITEMS	A enumeração listou todas as chaves disponíveis.

NCryptExportKey com o provedor de armazenamento de chaves (KSP)

A função NCryptExportKey exporta uma chave do KSP para um BLOB de memória. Essa função é compatível somente com a exportação de chaves públicas.

Parâmetros

hKey [in]

O identificador da chave a ser exportada.

hExportKey [in, unused]

AWS CloudHSM O Key Storage Provider (KSP) não usa esse parâmetro.

pszBlobType [in]

Uma string Unicode terminada em nulo que especifica o tipo a ser exportado. BLOB AWS CloudHSM O Key Storage Provider (KSP) suporta os seguintes valores:

Valor	Significado
BCRYPT_RSAPUBLIC_BLOB	Exporta uma chave pública do RSA. O buffer pbOutput contém uma estrutura BCRYPT_RSAKEY_BLOB seguida pelos principais dados.
BCRYPT_ECCPUBLIC_BLOB	Exporta uma chave pública do ECC. O buffer pbOutput contém uma estrutura BCRYPT_ECCKEY_BLOB seguida pelos principais dados.

pParameterList [in, unused]

AWS CloudHSM O Key Storage Provider (KSP) não usa esse parâmetro.

pbOutput [out, optional]

Um endereço de buffer para armazenar o BLOB de chaves. Especifique o tamanho do buffer usando `cbOutput`. Se definida como `NULL`, a função armazena o tamanho necessário (em bytes) no `DWORD` apontado por `pcbResult`.

cbOutput [in]

O tamanho em bytes do buffer `pbOutput`.

pcbResult [out]

Um endereço de variável `DWORD` que armazena o número de bytes copiados para o buffer `pbOutput`. Se `pbOutput` for `NULL`, a função armazenará o tamanho de buffer necessário em bytes.

dwFlags [in]

Sinalizadores que modificam o funcionamento da função. É possível usar nenhum ou o seguinte:

Valor	Significado
<code>NCRYPT_SILENT_FLAG</code>	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
<code>ERROR_SUCCESS</code>	A operação foi concluída com êxito.
<code>NTE_INVALID_PARAMETER</code>	Um ou mais parâmetros não são válidos.
<code>NTE_FAIL</code>	A operação não foi concluída.
<code>NTE_INVALID_HANDLE</code>	O identificador em <code>hProvider</code> não é válido.
<code>NTE_BAD_FLAGS</code>	O parâmetro <code>dwFlags</code> contém um valor inválido.

Código de retorno	Description
NTE_BAD_KEY_STATE	O estado da chave não é válido.
NTE_NOT_SUPPORTED	O parâmetro <code>pszBlobType</code> ou <code>dwFlags</code> contém um valor não suportado.
STATUS_INTERNAL_ERROR	Ocorreu um erro interno durante a operação.

NCryptSignHash com o provedor de armazenamento de chaves (KSP)

A função `NCryptSignHash` cria uma assinatura de um valor de hash.

Parâmetros

`hKey` [in]

O identificador da chave a ser usada para assinar o hash.

`pPaddingInfo` [in, optional]

Um ponteiro para uma estrutura contendo informações de preenchimento. O tipo de estrutura depende do valor de `dwFlags`. Use esse parâmetro somente com chaves assimétricas; defina como `NULL` para outros tipos de chave.

`pbHashValue` [in]

Um ponteiro para um buffer contendo o valor de hash a ser assinado. Especifique o tamanho do buffer usando `cbHashValue`.

`cbHashValue` [in]

O tamanho em bytes do buffer atual de `pbHashValue`.

`pbSignature` [out]

O endereço de um buffer para armazenar a assinatura. Especifique o tamanho do buffer usando `cbSignature`.

Para determinar o tamanho do buffer necessário, defina esse parâmetro como `NULL`. A função armazena o tamanho necessário (em bytes) no local apontado por `pcbResult`.

cbSignature [in]

O tamanho em bytes do buffer pbSignature. A função ignora esse parâmetro se pbSignature for NULL.

pcbResult [out]

Um ponteiro para uma variável DWORD que armazena o número de bytes copiados para o buffer pbSignature.

Se pbSignature for NULL, isso armazenará o tamanho necessário, em bytes.

dwFlags [in]

Sinalizadores para modificar o comportamento da função. Os sinalizadores permitidos dependem do seu tipo de chave. Use um destes valores:

Valor	Significado
BCRYPT_PAD_PKCS1	Usa o esquema PKCS1 de preenchimento. Defina pPaddingInfo para apontar para uma estrutura BCRYPT_PKCS1_PADDING_INFO .
BCRYPT_PAD_PSS	Usa o esquema de preenchimento do Probabilistic Signature Scheme (PSS). Defina o parâmetro pPaddingInfo para apontar para uma estrutura BCRYPT_PSS_PADDING_INFO .
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.

Código de retorno	Description
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_INVALID_HANDLE	O identificador em hKey não é válido.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.
NTE_BUFFER_TOO_SMALL	O parâmetro pcbOutput é muito pequeno para valores de retorno.
NTE_BAD_KEY_STATE	O estado da chave não é válido.
NTE_INTERNAL_ERROR	Ocorreu um erro interno ao assinar o hash.

NCryptVerifySignature com o provedor de armazenamento de chaves (KSP)

A função `NCryptVerifySignature` confirma se uma assinatura corresponde a um hash especificado.

Parâmetros

`hKey [in]`

O identificador da chave a ser usada para descriptografar a assinatura. Você deve usar a parte da chave pública do par de chaves que foi usada para assinar os dados com [NCryptSignHash](#).

`pPaddingInfo [in, optional]`

Um ponteiro para uma estrutura contendo informações de preenchimento. O tipo de estrutura depende do valor de `dwFlags`. Use esse parâmetro somente com chaves assimétricas; defina como `NULL` para outros tipos de chave.

`pbHashValue [in]`

Um ponteiro para um buffer contendo o valor de hash a ser assinado. Especifique o tamanho do buffer usando `cbHashValue`.

cbHashValue [in]

O tamanho em bytes do buffer pbHashValue.

pbSignature [out]

O endereço de um buffer contendo o hash assinado dos dados. Use [NCryptSignHash](#) para criar essa assinatura. Especifique o tamanho do buffer usando cbSignature.

cbSignature [in]

O tamanho em bytes do buffer pbSignature. Use [NCryptSignHash](#) para criar a assinatura.

dwFlags [in]

Sinalizadores para modificar o comportamento da função. Os sinalizadores permitidos dependem do seu tipo de chave. Use um destes valores:

Valor	Significado
NCRYPT_PAD_ _BANDEIRA PKCS1	Indica o PKCS1 preenchimento usado na assinatura. Defina pPaddingInfo para apontar para uma estrutura BCRYPT_PKCS1_PADDING_INFO .
NCRYPT_PAD_PSS_FLAG	Indica que a assinatura usou o preenchimento Probabilistic Signature Scheme (PSS). Defina pPaddingInfo para apontar para uma estrutura BCRYPT_PSS_PADDING_INFO .
NCRYPT_SILENT_FLAG	Esse sinalizador não tem efeito.

Valor de retorno

A função retorna um código de status para indicar êxito ou falha.

Os códigos de retorno comuns incluem:

Código de retorno	Description
ERROR_SUCCESS	A operação foi concluída com êxito.
NTE_INVALID_PARAMETER	Um ou mais parâmetros não são válidos.
NTE_FAIL	A operação não foi concluída.
NTE_INVALID_HANDLE	O identificador em hKey não é válido.
NTE_BAD_FLAGS	O parâmetro dwFlags contém um valor inválido.
NTE_BAD_SIGNATURE	A assinatura não foi verificada.
NTE_BAD_KEY_STATE	O estado da chave não é válido.
NTE_INTERNAL_ERROR	Ocorreu um erro interno ao verificar a assinatura.

Configurações avançadas do KSP para AWS CloudHSM

O AWS CloudHSM Key Storage Provider (KSP) inclui a seguinte configuração avançada, que não faz parte das configurações gerais que a maioria dos clientes utiliza. Essas configurações fornecem recursos adicionais.

- [SDK3 modo de compatibilidade para KSP](#)

SDK3 modo de compatibilidade do Key Storage Provider (KSP) para AWS CloudHSM

O Provedor de armazenamento de chaves (KSP) implementa diferentes abordagens para a interação de chaves do HSM:

- Client SDK 5: oferece comunicação direta com as chaves armazenadas no HSM, eliminando a necessidade de arquivos de referência locais
- Client SDK 3: mantém arquivos locais no servidor Windows que atuam como referências às chaves armazenadas no HSM, usando esses arquivos para facilitar as operações de chaves

Para clientes migrando do Client SDK 3 para o Client SDK 5, ativar a opção do modo de SDK3 compatibilidade suporta operações usando arquivos de referência de chave existentes, preservando a arquitetura subjacente de armazenamento de chaves do HSM.

Ativar o modo de SDK3 compatibilidade

Windows

Para ativar o modo de SDK3 compatibilidade do Key Storage Provider (KSP) para Client SDK 5 no Windows

- Você pode usar o comando a seguir para ativar o modo de SDK3 compatibilidade:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --enable-  
sdk3-compatibility-mode
```

Desativar o modo de SDK3 compatibilidade

Windows

Para desativar o modo de SDK3 compatibilidade do Key Storage Provider (KSP) para Client SDK 5 no Windows

- Você pode usar o comando a seguir para desativar o modo de SDK3 compatibilidade:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --disable-  
sdk3-compatibility-mode
```

Provedor de JCE para AWS CloudHSM Client SDK 5

O provedor AWS CloudHSM JCE é uma implementação de provedor criada a partir da estrutura do provedor Java Cryptographic Extension (JCE). O JCE permite que você execute operações criptográficas usando o Java Development Kit (JDK). Neste guia, o provedor AWS CloudHSM JCE às vezes é chamado de provedor JCE. Use o provedor JCE e o JDK para transferir operações criptográficas para o HSM. Para solução de problemas, consulte [Problemas conhecidos do JCE SDK para AWS CloudHSM](#).

Para obter mais informações sobre como usar Client SDK 3, consulte [Usando a versão anterior do SDK para trabalhar com AWS CloudHSM](#).

Tópicos

- [Instale o provedor JCE para o AWS CloudHSM Client SDK 5](#)
- [Tipos de chaves compatíveis com o provedor JCE para o AWS CloudHSM Client SDK 5](#)
- [Fundamentos do gerenciamento de chaves no provedor de JCE para o AWS CloudHSM Client SDK 5](#)
- [Mecanismos compatíveis com o provedor JCE para o AWS CloudHSM Client SDK 5](#)
- [Atributos de chave Java compatíveis com o AWS CloudHSM Client SDK 5](#)
- [Exemplos de código para a biblioteca AWS CloudHSM de software para Java for Client SDK 5](#)
- [AWS CloudHSM Javadocs, provedor de JCE](#)
- [AWS CloudHSM KeyStore Classe Java para Client SDK 5](#)
- [Configurações avançadas para AWS CloudHSM JCE for Client SDK 5](#)

Instale o provedor JCE para o AWS CloudHSM Client SDK 5

O provedor JCE para o AWS CloudHSM Client SDK 5 é compatível com OpenJDK 8, OpenJDK 11, OpenJDK 17, OpenJDK 21 e OpenJDK 25. Você pode baixar os dois no site do [OpenJDK](#).

Use as seções a seguir para instalar e fornecer credenciais ao provedor.

Note

Para executar um único cluster HSM com o Client SDK 5, você deve primeiro gerenciar as configurações de durabilidade da chave do cliente configurando `disable_key_availability_check` como `True`. Para obter mais informações, consulte [Sincronização de chave](#) e a [ferramenta de configuração do Client SDK 5](#).

Tópicos

- [Etapa 1: instalar o provedor JCE](#)
- [Etapa 2: fornecer credenciais ao provedor JCE](#)

Etapa 1: instalar o provedor JCE

1. Use os seguintes comandos para fazer download e instalar o provedor JCE.

Amazon Linux 2023

Instale o provedor JCE para Amazon Linux 2023 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-jce-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.amzn2023.x86_64.rpm
```

Instale o provedor JCE para Amazon Linux 2023 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-jce-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Instale o provedor JCE para Amazon Linux 2 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el7.x86_64.rpm
```

Instale o provedor JCE para Amazon Linux 2 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-jce-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

Instale o provedor JCE para RHEL 10 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-jce-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el10.x86_64.rpm
```

Instale o provedor JCE para RHEL 10 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-jce-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

Instale o provedor JCE para RHEL 9 (9.2+) na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-jce-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el9.x86_64.rpm
```

Instale o provedor JCE para RHEL 9 (9.2+) na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-jce-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

Instale o provedor JCE para RHEL 8 na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-jce-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el8.x86_64.rpm
```

Instale o provedor JCE para RHEL 8 na ARM64 arquitetura:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-jce-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el8.aarch64.rpm
```

Ubuntu 24.04 LTS

Instale o provedor JCE para Ubuntu 24.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-jce_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u24.04_amd64.deb
```

Instale o provedor JCE para Ubuntu 24.04 LTS na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-jce_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Instale o provedor JCE para Ubuntu 22.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-jce_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u22.04_amd64.deb
```

Instale o provedor JCE para o Ubuntu 22.04 LTS na arquitetura: ARM64

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-jce_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u22.04_arm64.deb
```

Windows Server

Instale o provedor JCE para Windows Server na arquitetura x86_64, abra PowerShell como administrador e execute o seguinte comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMJCE-latest.msi -Outfile C:\AWSCloudHSMJCE-latest.msi
```

```
PS C:\> Start-Process msixexec.exe -ArgumentList '/i C:\AWSCloudHSMJCE-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

2. Bootstrap do Client SDK 5. Para obter mais informações sobre ações de bootstrap, consulte [Bootstrap o Client SDK](#).
3. Localize os seguintes arquivos do provedor JCE:

Linux

- /opt/cloudhsm/java/cloudhsm-*<version>*.jar
- /opt/cloudhsm/bin/configure-jce
- /opt/cloudhsm/bin/jce-info

Windows

- C:\Program Files\Amazon\CloudHSM\java\cloudhsm-*<version>*.jar>
- C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe
- C:\Program Files\Amazon\CloudHSM\bin\jce_info.exe

Etapa 2: fornecer credenciais ao provedor JCE

Antes que seu aplicativo Java possa usar um HSM, o HSM precisa primeiro autenticar o aplicativo. HSMs autentique usando um método de login explícito ou implícito.

Login explícito AWS CloudHSM esse método permite que você forneça credenciais do diretamente no aplicativo. Ele usa o método [AuthProvider](#), em que você passa no nome do usuário, a senha e o ID da partição do HSM do usuário CU. Para obter mais informações, consulte [Exemplo de login em um código HSM](#).

Login implícito AWS CloudHSM esse método permite que você defina as credenciais do em um novo arquivo de propriedades, as propriedades do sistema, ou como variáveis de ambiente.

- Propriedades do sistema: defina as credenciais por meio das propriedades do sistema ao executar seu aplicativo. Os exemplos a seguir mostram duas formas diferentes de fazer isso:

Linux

```
$ java -DHSM_USER=<HSM user name> -DHSM_PASSWORD=<password>
```

```
System.setProperty("HSM_USER", "<HSM user name>");  
System.setProperty("HSM_PASSWORD", "<password>");
```

Windows

```
PS C:\> java -DHSM_USER=<HSM user name> -DHSM_PASSWORD=<password>
```

```
System.setProperty("HSM_USER", "<HSM user name>");  
System.setProperty("HSM_PASSWORD", "<password>");
```

- Variáveis de ambiente: defina as credenciais como variáveis de ambiente.

Note

Ao definir variáveis de ambiente, você deve escapar de quaisquer caracteres especiais que possam ser interpretados pelo seu shell.

Linux

```
$ export HSM_USER=<HSM user name>  
$ export HSM_PASSWORD=<password>
```

Windows

```
PS C:\> $Env:HSM_USER="<HSM user name>"
PS C:\> $Env:HSM_PASSWORD="<password>"
```

As credenciais talvez não estejam disponíveis se o aplicativo não fornecê-las ou se você tentar uma operação antes que o HSM autentique a sessão. Nesses casos, a biblioteca de software do CloudHSM para Java procura as credenciais na seguinte ordem:

1. Propriedades do sistema
2. Variáveis de ambiente

Tipos de chaves compatíveis com o provedor JCE para o AWS CloudHSM Client SDK 5

A biblioteca AWS CloudHSM de software para Java permite que você gere os seguintes tipos de chaves.

Tipo de chave	Description
AES	Gere chaves AES de 128, 192 e 256 bits.
DES triplo (3DES, DESede)	Gere uma chave DES tripla de 192 bits. *
EC	Gere pares de chaves EC com as curvas NIST secp224r1 (P-224), secp256r1 (P-256), secp256k1 (Blockchain), secp384r1 (P-384) e secp521r1 (P-521).
GENERIC_SECRET	Gerar chaves genéricas de 1 a 800 bytes.
HMAC	Suporte de hash para SHA1, SHA224,, SHA256 SHA384, SHA512.
RSA	Gera chaves RSA de 2.048 a 4.096 bits, em incrementos de 256 bits.

* De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Fundamentos do gerenciamento de chaves no provedor de JCE para o AWS CloudHSM Client SDK 5

Os conceitos básicos do gerenciamento de chaves no provedor JCE envolvem a importação e a exportação, o carregamento por identificador ou a exclusão de chaves. Para obter mais informações sobre como gerenciar chaves, consulte [Gerenciar chaves](#) no exemplo de código.

Você também pode encontrar mais amostras de código de provedor JCE em [Exemplos de código](#).

Mecanismos compatíveis com o provedor JCE para o AWS CloudHSM Client SDK 5

Este tópico fornece informações sobre os mecanismos suportados pelo provedor JCE com o AWS CloudHSM Client SDK 5. Para obter informações sobre as interfaces e classes de mecanismo da Java Cryptography Architecture (JCA) suportadas pelo AWS CloudHSM, consulte os tópicos a seguir.

Tópicos

- [Gere funções de chave e par de chaves](#)
- [Funções de cifra](#)
- [Funções de assinatura e verificação](#)
- [Funções de resumo](#)
- [Funções de código de autenticação de mensagens por hash \(HMAC\)](#)
- [Funções de código de autenticação de mensagens baseadas em cifras \(CMAC\)](#)
- [Funções-chave do contrato](#)
- [Converta chaves em especificações de chave usando fábricas de chaves](#)
- [Anotações do mecanismo](#)

Gere funções de chave e par de chaves

A biblioteca AWS CloudHSM de software para Java permite que você use as seguintes operações para gerar funções de chave e par de chaves.

- RSA
- EC
- AES
- DESede (Triple DES) consulte a observação [1](#)
- GenericSecret

Funções de cifra

A biblioteca AWS CloudHSM de software para Java suporta as seguintes combinações de algoritmo, modo e preenchimento.

Algoritmo	Modo	Padding	Observações
AES	CBC	AES/CBC/NoPadding AES/CBC/PKCS5Padding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE . Implementa Cipher.UNWRAP_MODE for AES/CBC NoPadding
AES	ECB	AES/ECB/PKCS5Padding AES/ECB/NoPadding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE .
AES	CTR	AES/CTR/NoPadding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE .

Algoritmo	Modo	Padding	Observações
AES	GCM	AES/GCM/NoPadding	Implementa Cipher.WRAP_MODE , Cipher.UNWRAP_MODE , Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE . Ao executar a criptografia AES-GCM, o HSM ignora o vetor de inicialização (IV) na solicitação e usa um IV que ele mesmo gera. Quando a operação for concluída, você deverá chamar Cipher.getIV() para obter o IV.
AESWrap	ECB	AESWrap/ECB/NoPadding AESWrap/ECB/PKCS5Padding AESWrap/ECB/ZeroPadding	Implementa Cipher.WRAP_MODE e Cipher.UNWRAP_MODE .

Algoritmo	Modo	Padding	Observações
DESede (DES triplo)	CBC	DESede/CBC/ PKCS5Padding DESede/CBC/ NoPadding	Implementa Cipher.EN CRYPT_MODE e Cipher.DE CRYPT_MODE . Consulte a nota 1 abaixo para ver uma mudança futura.
DESede (DES triplo)	ECB	DESede/ECB/ NoPadding DESede/ECB/ PKCS5Padding	Implementa Cipher.EN CRYPT_MODE e Cipher.DE CRYPT_MODE . Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações
RSA	ECB	RSA/ECB/P KCS1Padding consulte a observação 1	Implementa Cipher.WR AP_MODE , Cipher.UN WRAP_MODE , Cipher.EN CRYPT_MODE e Cipher.DE CRYPT_MODE .
		RSA/ECB/0 AEPPadding	
		RSA/ECB/0 AEPWithSH A-1ANDMGF 1Padding	
		RSA/ECB/0 AEPWithSH A-224ANDM GF1Padding	
		RSA/ECB/0 AEPWithSH A-256ANDM GF1Padding	
		RSA/ECB/0 AEPWithSH A-384ANDM GF1Padding	
		RSA/ECB/0 AEPWithSH A-512ANDM GF1Padding	

Algoritmo	Modo	Padding	Observações
RSA	ECB	RSA/ECB/NoPadding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE .
RSAAESWrap	ECB	RSAAESWrap/ECB/OAEP RSAAESWrap/ECB/OAEPWithSHA-1ANDMGF1Padding RSAAESWrap/ECB/OAEPWithSHA-224ANDMGF1Padding RSAAESWrap/ECB/OAEPWithSHA-256ANDMGF1Padding RSAAESWrap/ECB/OAEPWithSHA-384ANDMGF1Padding RSAAESWrap/ECB/OAEPWithSHA-512ANDMGF1Padding	Implementa Cipher.WRAP_MODE e Cipher.UNWRAP_MODE .

Funções de assinatura e verificação

A biblioteca AWS CloudHSM de software para Java suporta os seguintes tipos de assinatura e verificação. Com o Client SDK 5 e algoritmos de assinatura com hashing, os dados são codificados localmente no software antes de serem enviados ao HSM para assinatura/verificação. Isso significa que não há limite no tamanho dos dados que podem ser criptografados pelo SDK.

Tipos de assinatura RSA

- `NONEwithRSA`
- `RSASSA-PSS`
- `SHA1withRSA`
- `SHA1withRSA/PSS`
- `SHA1withRSAandMGF1`
- `SHA224withRSA`
- `SHA224withRSAandMGF1`
- `SHA224withRSA/PSS`
- `SHA256withRSA`
- `SHA256withRSAandMGF1`
- `SHA256withRSA/PSS`
- `SHA384withRSA`
- `SHA384withRSAandMGF1`
- `SHA384withRSA/PSS`
- `SHA512withRSA`
- `SHA512withRSAandMGF1`
- `SHA512withRSA/PSS`

Tipos de assinatura ECDSA

- `NONEwithECDSA`
- `SHA1withECDSA`
- `SHA224withECDSA`

- SHA256withECDSA
- SHA384withECDSA
- SHA512withECDSA

Funções de resumo

A biblioteca AWS CloudHSM de software para Java suporta os seguintes resumos de mensagens. Com o Client SDK 5, os dados são codificados localmente no software. Isso significa que não há limite no tamanho dos dados que podem ser criptografados pelo SDK.

- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512

Funções de código de autenticação de mensagens por hash (HMAC)

A biblioteca AWS CloudHSM de software para Java suporta os seguintes algoritmos HMAC.

- HmacSHA1 (Tamanho máximo de dados em bytes: 16288)
- HmacSHA224 (Tamanho máximo de dados em bytes: 16256)
- HmacSHA256 (Tamanho máximo de dados em bytes: 16288)
- HmacSHA384 (Tamanho máximo de dados em bytes: 16224)
- HmacSHA512 (Tamanho máximo de dados em bytes: 16224)

Funções de código de autenticação de mensagens baseadas em cifras (CMAC)

CMACs (códigos de autenticação de mensagens baseados em cifras) crie códigos de autenticação de mensagens (MACs) usando uma cifra de bloco e uma chave secreta. Eles diferem HMACs porque usam um método de chave simétrica de bloco para o, MACs em vez de um método de hash.

A biblioteca AWS CloudHSM de software para Java suporta os seguintes algoritmos CMAC.

- AESCMAC

Funções-chave do contrato

A biblioteca AWS CloudHSM de software para Java suporta ECDH com Key Derivation Functions (KDF). Há suporte para os seguintes tipos de KDF:

- `ECDHwithX963SHA1KDF` Suporta o algoritmo X9.63 KDF SHA1 [2](#)
- `ECDHwithX963SHA224KDF` Suporta o algoritmo X9.63 KDF SHA224 [2](#)
- `ECDHwithX963SHA256KDF` Suporta o algoritmo X9.63 KDF SHA256 [2](#)
- `ECDHwithX963SHA384KDF` Suporta o algoritmo X9.63 KDF SHA384 [2](#)
- `ECDHwithX963SHA512KDF` Suporta o algoritmo X9.63 KDF SHA512 [2](#)

Converta chaves em especificações de chave usando fábricas de chaves

Você pode usar fábricas de chaves para converter chaves em especificações principais. AWS CloudHSM tem dois tipos de fábricas principais para a JCE:

`SecretKeyFactory`: usado para importar ou derivar chaves simétricas. Usando `SecretKeyFactory`, você pode passar uma chave compatível ou uma compatível `KeySpec` para importar ou derivar chaves simétricas. AWS CloudHSM A seguir estão as especificações suportadas para `KeyFactory`:

- As seguintes [KeySpec](#) classes `SecretKeyFactory` do `generateSecret` método For são suportadas:
 - `KeyAttributesMap` pode ser usado para importar bytes de chave com atributos adicionais como uma chave do CloudHSM. Um exemplo pode ser encontrado [aqui](#)
 - [SecretKeySpec](#) pode ser usado para importar uma especificação de chave simétrica como uma chave do CloudHSM.
 - `AesCmacKdfParameterSpec` pode ser usado para derivar chaves simétricas usando outra chave AES do CloudHSM.

Note

`SecretKeyFactory` O `translateKey` método de usa qualquer chave que implemente a interface da [chave](#).

KeyFactory: usado para importar chaves assimétricas. Usando KeyFactory, você pode passar uma chave compatível ou compatível KeySpec para importar uma chave assimétrica. AWS CloudHSM Para obter mais informações, consulte os seguintes recursos:

- Para KeyFactory o generatePublic método de For, [KeySpec](#)as seguintes classes são suportadas:
- KeyAttributesMap CloudHSM para RSA e EC, incluindo: KeyTypes
 - KeyAttributesMap CloudHSM para o público da RSA e da EC. KeyTypes Um exemplo pode ser encontrado [aqui](#)
 - [X509 EncodedKeySpec](#) para chave pública RSA e EC
 - [RSAPublicKeySpec](#) para chave pública RSA
 - [ECPublicKeySpec](#) para EC Public Key
- Para KeyFactory o generatePrivate método de For, [KeySpec](#)as seguintes classes são suportadas:
- KeyAttributesMap CloudHSM para RSA e EC, incluindo: KeyTypes
 - KeyAttributesMap CloudHSM para o público da RSA e da EC. KeyTypes Um exemplo pode ser encontrado [aqui](#)
 - [PKCS8EncodedKeySpec](#) para chaves privadas EC e RSA
 - [RSAPrivateCrtKeySpec](#) para chave privada RSA
 - [ECPrivateKeySpec](#) para chave privada EC

KeyFactoryO translateKey método de For usa qualquer chave que implemente a [interface chave](#).

Anotações do mecanismo

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

[2] As funções de derivação de chaves (KDFs) estão especificadas na [Publicação Especial 800-56A do NIST](#), Revisão 3.

Atributos de chave Java compatíveis com o AWS CloudHSM Client SDK 5

Este tópico fornece informações sobre os atributos de chave Java compatíveis com o AWS CloudHSM Client SDK 5. Este tópico descreve como usar uma extensão proprietária para o provedor

JCE para definir atributos de chave. Use essa extensão para definir atributos de chave compatíveis e seus valores durante estas operações:

- Geração de chaves
- Importação de chaves

Para obter exemplos de como usar os principais atributos, consulte [the section called “Exemplos de código”](#).

Tópicos

- [Noções básicas sobre atributos](#)
- [Atributos do compatíveis](#)
- [Definir atributos para uma chave](#)

Noções básicas sobre atributos

Use atributos de chave para especificar quais ações são permitidas em objetos de chave, incluindo chaves públicas, privadas ou secretas. Defina os atributos e valores de chave durante as operações de criação de objetos de chave.

A Java Cryptography Extension (JCE) não especifica como você deve definir valores em atributos de chave, portanto, a maioria das ações foi permitida por padrão. Em contrapartida, o padrão PKCS #11 define um conjunto de atributos abrangente com padrões mais restritivos. Começando com o provedor JCE 3.1, AWS CloudHSM fornece uma extensão proprietária que permite definir valores mais restritivos para atributos comumente usados.

Atributos do compatíveis

É possível definir valores para atributos listados na tabela abaixo. Como melhor prática, defina valores somente para os atributos que deseja tornar restritivos. Se você não especificar um valor, AWS CloudHSM usa o valor padrão especificado na tabela abaixo. Uma célula vazia nas colunas Valor padrão indica que não há nenhum valor padrão específico atribuído ao atributo.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
DECRYPT	TRUE		TRUE	True indica que você pode usar a chave para descriptografar qualquer buffer. Geralmente, você define isso como FALSE para uma chave cujo WRAP esteja definido como true.
DERIVE				Permite que uma chave seja usada para obter outras chaves.
ENCRYPT	TRUE	TRUE		True indica que você pode usar a chave para criptografar qualquer buffer.
EXTRACTABLE	TRUE		TRUE	True indica que você pode exportar essa chave para fora do HSM.
ID				Um valor definido pelo usuário usado

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
				para identificar a chave.
KEY_TYPE				Usado para identificar o tipo de chave (AES DESede, segredo genérico, EC ou RSA).
LABEL				Uma string definida pelo usuário que permite que você identifique convenientemente as chaves no seu HSM. Para seguir as melhores práticas, use um rótulo exclusivo para cada chave para que seja mais fácil encontrá-la posteriormente.
LOCAL				Indica uma chave gerada pelo HSM.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
OBJECT_CLASS				Usado para identificar a classe de objeto de uma chave (SecretKey PublicKey ou PrivateKey).
PRIVATE	TRUE	TRUE	TRUE	True indica que um usuário pode não acessar a chave até que o usuário seja autenticado. Para maior clareza, os usuários não podem acessar nenhuma chave AWS CloudHSM até serem autenticados, mesmo que esse atributo esteja definido como FALSE.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
SIGN	TRUE		TRUE	True indica que você pode usar a chave para assinar um resumo de mensagens. Geralmente, é definido como FALSE para chaves públicas e para chaves privadas arquivadas.
SIZE				Um atributo que define o tamanho de uma chave. Para obter mais detalhes sobre os tamanhos de chave compatíveis, consulte Supported mechanisms for Client SDK 5 .

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
TOKEN	FALSE	FALSE	FALSE	Uma chave permanente que é replicada HSMs em todo o cluster e incluída nos backups. TOKEN = FALSE implica uma chave efêmera que é apagada automaticamente quando a conexão com o HSM é interrompida ou é feito logout.
UNWRAP	TRUE		TRUE	True indica que você pode usar a chave para desencapsular (importar) outra chave.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
VERIFY	TRUE	TRUE		True indica que você pode usar a chave para verificar uma assinatura. Isso geralmente é definido como FALSE para chaves privadas.
WRAP	TRUE	TRUE		True indica que você pode usar a chave para encapsular outra chave. Geralmente, isso é definido como FALSE para chaves privadas.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
WRAP_WITH _TRUSTED	FALSE		FALSE	Verdadeiro indica que uma chave só pode ser encapsulada e desencapsulada com chaves que tenham o atributo TRUSTED definido como verdadeiro. Depois que uma chave tem o WRAP_WITH_TRUSTED definido como verdadeiro, esse atributo é somente para leitura e não pode ser definido como falso. Para ler sobre o agrupamento confiável, consulte Using trusted keys to control key unwraps .

Note

Você obterá maior suporte para atributos na biblioteca PKCS #11. Para obter mais informações, consulte [Atributos PKCS #11 compatíveis](#).

Definir atributos para uma chave

O `KeyAttributesMap` é um objeto semelhante ao Mapa Java, que pode ser usado para definir valores de atributos para objetos de chave. Os métodos para a função `KeyAttributesMap` funcionam de forma semelhante aos métodos usados na manipulação de mapa Java.

Para definir valores personalizados, existem duas opções:

- Usar os métodos listados na tabela a seguir
- Usar padrões do construtor demonstrados posteriormente nesse documento.

Os objetos de mapa de atributos oferecem suporte para os seguintes métodos para definir atributos:

Operation	Valor de retorno	Método KeyAttributesMap
Obter o valor de um atributo de chave para uma chave existente	Objeto (contendo o valor) ou nulo	<code>get(keyAttribute)</code>
Preencher o valor de um atributo de chave	O valor anterior associado a um atributo de chave, ou nulo se não houver mapeamento para um atributo de chave	<code>put(keyAttribute, valor)</code>
Preencher valores para múltiplos atributos de chave	N/D	<code>putAll () keyAttributesMap</code>
Remover um par de valor-chave do mapa de atributos	O valor anterior associado a um atributo de chave, ou nulo se não houver mapeamento para um atributo de chave	<code>remove(keyAttribute)</code>

 Note

Todos os atributos que você não especificar explicitamente serão definidos como os padrões listados na tabela anterior em [the section called “Atributos do compatíveis”](#).

Definir atributos para um par de chaves

Use a classe Java `KeyPairAttributesMap` para manipular atributos de chave para um par de chaves. O `KeyPairAttributesMap` encapsula dois objetos `KeyAttributesMap`; um para uma chave pública e outro para uma chave privada.

Para definir atributos individuais para a chave pública e privada separadamente, é possível usar o método `put()` no objeto de mapa `KeyAttributes` correspondente para essa chave. Use o método `getPublic()` para recuperar o mapa de atributos para a chave pública e use `getPrivate()` para recuperar o mapa de atributos para a chave privada. Preencha o valor de múltiplos atributos de chave para os pares de chaves públicas e privadas usando `putAll()` com um mapa de atributos de um par de chaves como argumento.

Exemplos de código para a biblioteca AWS CloudHSM de software para Java for Client SDK 5

Este tópico fornece recursos e informações sobre exemplos de código Java para o Client SDK 5 do AWS CloudHSM .

Pré-requisitos

Antes de executar as amostras, você deve configurar seu ambiente:

- Instale e configure o provedor [Java Cryptographic Extension \(JCE – extensão de criptografia Java\)](#).
- Configure um [nome de usuário e senha de HSM](#) válidos. As permissões do usuário de criptografia (CU) são suficientes para essas tarefas. O aplicativo usa essas credenciais para fazer login no HSM em cada exemplo.
- Decida como fornecer credenciais ao provedor [JCE](#).

Exemplos de código

Os exemplos de código a seguir mostram como usar o [provedor JCE AWS CloudHSM](#) para realizar tarefas básicas. Mais exemplos de código estão disponíveis em [GitHub](#).

- [Log in to an HSM](#)
- [Gerenciar chaves](#)
- [Gerar chaves simétricas](#)
- [Gerar chaves assimétricas](#)
- [Encrypt and decrypt with AES-GCM](#)
- [Encrypt and decrypt with AES-CTR](#)
- [Criptografe e descriptografe com -ECB, veja a nota DESede 1](#)
- [Assinatura e verificação com chaves RSA](#)
- [Assinatura e verificação com chaves EC](#)
- [Use supported key attributes](#)
- [Use the CloudHSM key store](#)

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

AWS CloudHSM Javadocs, provedor de JCE

Use o Provedor JCE Javadocs para obter informações de uso sobre os tipos e métodos Java definidos no JCE do SDK do AWS CloudHSM. Para baixar os Javadocs mais recentes AWS CloudHSM, consulte a [AWS CloudHSM versão mais recente do SDK do cliente](#) seção na página Downloads.

Você pode importar Javadocs em um ambiente de desenvolvimento integrado (IDE) ou visualizá-los em um navegador da Web.

AWS CloudHSM KeyStore Classe Java para Client SDK 5

A AWS CloudHSM KeyStore classe fornece um armazenamento de PKCS12 chaves para fins especiais. Este repositório de chaves pode armazenar certificados junto com os seus dados de chave e correlacioná-los com os dados da chave armazenados no AWS CloudHSM. A AWS CloudHSM KeyStore classe implementa a KeyStore Service Provider Interface (SPI) da Java Cryptography Extension (JCE). Para obter mais informações sobre o usoKeyStore, consulte [Classe KeyStore](#).

 Note

Como os certificados são informações públicas e, para maximizar a capacidade de armazenamento de chaves criptográficas, AWS CloudHSM não oferece suporte ao armazenamento de certificados em HSMs.

Escolha o armazenamento de chaves apropriado para o AWS CloudHSM Client SDK 5

O provedor de AWS CloudHSM Java Cryptographic Extension (JCE) oferece um AWS CloudHSM para fins especiais. KeyStore A AWS CloudHSM KeyStore classe oferece suporte ao descarregamento de operações-chave para o HSM, ao armazenamento local de certificados e às operações baseadas em certificados.

Carregue o CloudHSM para fins especiais da seguinte forma: KeyStore

```
KeyStore ks = KeyStore.getInstance("CloudHSM")
```

Inicializar o AWS CloudHSM KeyStore Client SDK 5

Faça login AWS CloudHSM KeyStore da mesma forma que você faz login no provedor JCE. Você pode usar variáveis de ambiente ou o arquivo de propriedades do sistema e deve fazer login antes de começar a usar o CloudHSM KeyStore. Para obter um exemplo de login em um HSM usando o JCE, consulte [Login em um HSM](#).

Se desejar, você pode especificar uma senha para criptografar o PKCS12 arquivo local que contém os principais dados do armazenamento. Ao criar o AWS CloudHSM Keystore, você define a senha e a fornece ao usar os métodos load, set e get.

Instancie um novo objeto CloudHSM da seguinte forma: KeyStore

```
ks.load(null, null);
```

Grave dados de repositório de chaves em um arquivo usando o método store. A partir desse ponto, você pode carregar o repositório de chaves existente usando o método load com o arquivo de origem e a senha da seguinte forma:

```
ks.load(inputStream, password);
```

Use AWS CloudHSM KeyStore nosso SDK de AWS CloudHSM cliente 5

AWS CloudHSM KeyStore está em conformidade com a KeyStore especificação da [classe](#) JCE e fornece as seguintes funções.

- **load**

Carrega o repositório de chaves do fluxo de entrada fornecido. Se uma senha foi definida ao salvar o repositório de chaves, essa mesma senha deve ser fornecida para que o carregamento seja bem-sucedido. Defina ambos os parâmetros como null para inicializar um novo repositório de chaves vazio.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
ks.load(inputStream, password);
```

- **aliases**

Retorna uma enumeração dos nomes de alias de todas as entradas na instância de repositório de chaves dada. Os resultados incluem objetos armazenados localmente no PKCS12 arquivo e objetos residentes no HSM.

Código de exemplo:

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
for(Enumeration<String> entry = ks.aliases(); entry.hasMoreElements();) {
    String label = entry.nextElement();
    System.out.println(label);
}
```

- **containsalias**

Retorna true se o repositório de chaves tiver acesso a pelo menos um objeto com o alias especificado. O armazenamento de chaves verifica os objetos armazenados localmente no PKCS12 arquivo e os objetos residentes no HSM.

- **deleteEntry**

Exclui uma entrada de certificado do PKCS12 arquivo local. A exclusão de dados-chave armazenados em um HSM não é suportada usando o. AWS CloudHSM KeyStore Você pode excluir chaves usando o método `destroy` da interface [Destruível](#).

```
((Destroyable) key).destroy();
```

- **getCertificate**

Retorna o certificado associado a um alias, se disponível. Se o alias não existir ou fizer referência a um objeto que não for um certificado, a função retornará NULL.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");  
Certificate cert = ks.getCertificate(alias);
```

- **getCertificateAlias**

Retorna o nome (alias) da primeira entrada de repositório de chaves cujos dados correspondem ao certificado fornecido.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");  
String alias = ks.getCertificateAlias(cert);
```

- **getCertificateChain**

Retorna a cadeia de certificados associada ao alias fornecido. Se o alias não existir ou fizer referência a um objeto que não for um certificado, a função retornará NULL.

- **getCreationDate**

Retorna a data de criação da entrada identificada pelo alias fornecido. Se uma data de criação não estiver disponível, a função retornará a data em que o certificado se tornou válido.

- **getKey**

GetKey é passado para o HSM e retorna um objeto chave correspondente ao rótulo fornecido. Ao consultar getKey diretamente o HSM, ela pode ser usada para qualquer chave no HSM, independentemente de ter sido gerada pelo. KeyStore

```
Key key = ks.getKey(keyLabel, null);
```

- **isCertificateEntry**

Verifica se a entrada com o alias fornecido representa uma entrada de certificado.

- **isKeyEntry**

Verifica se a entrada com o alias fornecido representa uma entrada de chave. A ação pesquisa o alias no PKCS12 arquivo e no HSM.

- `setCertificateEntry`

Atribui o certificado fornecido ao alias fornecido. Se o alias fornecido já estiver sendo usado para identificar uma chave ou certificado, um `KeyStoreException` é lançado. Você pode usar o código JCE para obter o objeto chave e, em seguida, usar o `KeyStore SetKeyEntry` método para associar o certificado à chave.

- `setKeyEntry` com chave `byte[]`

No momento, essa API não é compatível com o Client SDK 5.

- `setKeyEntry` com objeto `Key`

Atribui a chave fornecida ao alias fornecido e armazena-a dentro do HSM. Se a chave ainda não existir dentro do HSM, ela será importada para o HSM como uma chave de sessão extraível.

Se o objeto `Key` for do tipo `PrivateKey`, ele deve ser acompanhado por uma cadeia de certificados correspondente.

Se o alias já existir, a `SetKeyEntry` chamada lança um `KeyStoreException` e impede que a chave seja substituída. Se a chave precisar ser substituída, use KMU ou JCE para esse fim.

- `engineSize`

Retorna o número de entradas no repositório de chaves.

- `store`

Armazena o armazenamento de chaves no fluxo de saída fornecido como um PKCS12 arquivo e o protege com a senha fornecida. Além disso, mantém todas as chaves carregadas (que são definidas usando chamadas `setKey`).

Configurações avançadas para AWS CloudHSM JCE for Client SDK 5

O provedor de AWS CloudHSM JCE inclui as seguintes configurações avançadas, que não fazem parte das configurações gerais que a maioria dos clientes utiliza.

- [Conectando-se a vários clusters](#)

- [Extração de chaves usando JCE](#)
- [Repita a configuração para JCE](#)

Conectando-se a vários AWS CloudHSM clusters com o provedor JCE

Essa configuração permite que uma única instância cliente se comunique com vários clusters do AWS CloudHSM. Comparado a ter uma única instância se comunicando apenas com um único cluster, esse pode ser um atributo de economia de custos para alguns casos de uso. A `CloudHsmProvider` classe é a implementação AWS CloudHSM da [classe Provider da Java Security](#). Cada instância dessa classe representa uma conexão com todo o AWS CloudHSM cluster. Você instancia essa classe e a adiciona à lista do provedor de segurança Java para poder interagir com ela usando classes JCE padrão.

O exemplo a seguir instancia essa classe e a adiciona à lista do provedor de segurança Java:

```
if (Security.getProvider(CloudHsmProvider.PROVIDER_NAME) == null) {  
    Security.addProvider(new CloudHsmProvider());  
}
```

`CloudHsmProvider` pode ser configurado de duas maneiras:

1. Configurar com arquivo (configuração padrão)
2. Configurar usando código

Os tópicos a seguir descrevem essas configurações e como se conectar a vários clusters.

Tópicos

- [Configurar a classe `CloudHsmProvider` do AWS CloudHSM com um arquivo \(configuração padrão\)](#)
- [Configurar a classe `CloudHsmProvider` do AWS CloudHSM usando código](#)
- [Conecte-se a vários AWS CloudHSM clusters](#)

Configurar a classe **`CloudHsmProvider`** do AWS CloudHSM com um arquivo (configuração padrão)

A forma padrão de configurar a AWS CloudHSM `CloudHsmProvider` classe é com um arquivo.

Quando você instancia `CloudHsmProvider` usando o construtor padrão, por padrão, ele procurará o arquivo de configuração no `/opt/cloudhsm/etc/cloudhsm-jce.cfg` caminho no Linux. Esse arquivo de configuração pode ser configurado usando `configure-jce`.

Um objeto criado usando o construtor padrão usará o nome padrão do provedor do CloudHSM CloudHSM. O nome do provedor é útil para interagir com o JCE para que ele saiba qual provedor usar para várias operações. Um exemplo de uso do nome do provedor CloudHSM para a operação Cipher é o seguinte:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", "CloudHSM");
```

Configurar a classe **CloudHsmProvider** do AWS CloudHSM usando código

A partir do Client SDK versão 5.8.0, você também pode configurar a AWS CloudHSM `CloudHsmProvider` classe usando o código Java. A maneira de fazer isso é usando um objeto de `CloudHsmProviderConfig` classe. Você pode criar esse objeto usando `CloudHsmProviderConfigBuilder`.

`CloudHsmProvider` tem outro construtor que pega o `CloudHsmProviderConfig` objeto, como mostra o exemplo a seguir.

Example

```
CloudHsmProviderConfig config = CloudHsmProviderConfig.builder()
    .withCluster(
        CloudHsmCluster.builder()
            .withHsmCAFilePath(hsmCAFilePath)

        .withClusterUniqueIdentifier("CloudHsmCluster1")
        .withServer(CloudHsmServer.builder().withHostIP(hostName).build())
        .build()
    ).build();
CloudHsmProvider provider = new CloudHsmProvider(config);
```

Neste exemplo, o nome do provedor JCE é `CloudHsmCluster1`. Este é o nome que a aplicação pode então usar para interagir com o JCE:

Example

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", "CloudHsmCluster1");
```

Como alternativa, os aplicativos também podem usar o objeto provedor criado acima para informar à JCE que deve usar esse provedor para a operação:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", provider);
```

Se um identificador exclusivo não for especificado com o método `withClusterUniqueIdentifier`, um nome de provedor gerado aleatoriamente será criado para você. Para obter esse identificador gerado aleatoriamente, os aplicativos podem chamar `provider.getName()` para obter o identificador.

Conecte-se a vários AWS CloudHSM clusters

Cada um `CloudHsmProvider` representa uma conexão com seu AWS CloudHSM cluster. Se quiser se comunicar com outro cluster do mesmo aplicativo, você pode criar outro objeto de `CloudHsmProvider` com configurações para seu outro cluster e interagir com esse outro cluster usando o objeto provedor ou usando o nome do provedor, conforme mostrado no exemplo a seguir.

Example

```
CloudHsmProviderConfig config = CloudHsmProviderConfig.builder()
    .withCluster(
        CloudHsmCluster.builder()
            .withHsmCAFilePath(hsmCAFilePath)

        .withClusterUniqueIdentifier("CloudHsmCluster1")
        .withServer(CloudHsmServer.builder().withHostIP(hostName).build())
        .build()
    ).build();
CloudHsmProvider provider1 = new CloudHsmProvider(config);

if (Security.getProvider(provider1.getName()) == null) {
    Security.addProvider(provider1);
}

CloudHsmProviderConfig config2 = CloudHsmProviderConfig.builder()
    .withCluster(
        CloudHsmCluster.builder()
            .withHsmCAFilePath(hsmCAFilePath2)

        .withClusterUniqueIdentifier("CloudHsmCluster2")
        .withServer(CloudHsmServer.builder().withHostIP(hostName2).build())
```

```
        .build())  
    .build();  
CloudHsmProvider provider2 = new CloudHsmProvider(config2);  
  
if (Security.getProvider(provider2.getName()) == null) {  
    Security.addProvider(provider2);  
}
```

Depois de configurar os dois provedores (os dois clusters) acima, você pode interagir com eles usando o objeto do provedor ou usando o nome do provedor.

Expandindo esse exemplo que mostra como falar com `cluster1`, você pode usar o exemplo a seguir para uma AES/GCM/NoPadding operação:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", provider1);
```

E no mesmo aplicativo para fazer a geração da chave “AES” no segundo cluster usando o nome do provedor, você também pode usar o seguinte exemplo:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", provider2.getName());
```

Extração de chaves usando JCE para AWS CloudHSM

A Java Cryptography Extension (JCE) usa uma arquitetura que permite que diferentes implementações de criptografia sejam conectadas. AWS CloudHSM envia um desses fornecedores de JCE que transfere operações criptográficas para o HSM. Para que a maioria dos outros provedores de JCE trabalhem com chaves armazenadas no AWS CloudHSM, eles devem extrair os bytes HSMs da chave em texto não criptografado na memória da sua máquina para serem usados. HSMs normalmente só permitem que as chaves sejam extraídas como objetos agrupados, não como texto claro. No entanto, para oferecer suporte a casos de uso de integração entre provedores, AWS CloudHSM permite uma opção de configuração opcional para permitir a extração dos bytes da chave em branco.

Important

A JCE transfere as operações para AWS CloudHSM sempre que o provedor do AWS CloudHSM é especificado ou um objeto chave é usado. AWS CloudHSM Você não precisa extrair as chaves de forma clara se você espera que sua operação ocorra dentro do HSM. A extração de chaves em texto não criptografado só é necessária quando seu aplicativo

não pode usar mecanismos seguros, como empacotar e desempacotar uma chave devido a restrições de uma biblioteca terceirizada ou de um provedor de JCE.

O provedor AWS CloudHSM JCE permite a extração de chaves públicas para funcionar com provedores JCE externos por padrão. Os seguintes métodos são sempre permitidos:

Classe	Método	Format (getEncoded)
EcPublicKey	getEncoded()	X.509
	getW()	N/D
RSAPublicChave	getEncoded()	X.509
	getPublicExponent()	N/D
CloudHsmRsaPrivateCrtKey	getPublicExponent()	N/D

O provedor AWS CloudHSM JCE não permite a extração de bytes de chave em branco para as chaves privadas ou secretas por padrão. Se seu caso de uso exigir isso, você pode habilitar a extração de bytes de chave em branco para chaves privadas ou secretas nas seguintes condições:

1. O EXTRACTABLE atributo para chaves privadas e secretas é definido como verdadeiro.
 - Por padrão, o EXTRACTABLE atributo para chaves privadas e secretas é definido como verdadeiro. Chaves EXTRACTABLE são chaves que podem ser exportadas para fora do HSM. Para obter mais informações, consulte Atributos Java compatíveis para o [Client SDK 5](#).
2. O WRAP_WITH_TRUSTED atributo para chaves privadas e secretas é definido como falso.
 - getEncoded, getPrivateExponent, e getS não podem ser usados com chaves privadas que não podem ser exportadas em branco. WRAP_WITH_TRUSTED não permite que suas chaves privadas sejam exportadas do HSM em branco. Para obter mais informações, consulte [Usando chaves confiáveis para controlar o desencapsulamento de chaves](#).

Permita que o provedor de JCE extraia segredos de chave privada do AWS CloudHSM

Use as etapas a seguir para permitir que o provedor de AWS CloudHSM JCE extraia seus segredos de chave privada.

⚠ Important

Essa alteração de configuração permite a extração de todos os bytes de EXTRACTABLE chave em branco do seu cluster HSM. Para maior segurança, você deve considerar o uso de [métodos de encapsulamento de chaves](#) para extrair a chave do HSM com segurança. Isso evita a extração não intencional dos seus bytes de chave do HSM.

1. Use os comandos a seguir para permitir que suas chaves privadas ou secretas sejam extraídas no JCE:

Linux

```
$ /opt/cloudhsm/bin/configure-jce --enable-clear-key-extraction-in-software
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --enable-clear-key-extraction-in-software
```

2. Depois de ativar a extração da chave em branco, os métodos a seguir são habilitados para extrair chaves privadas na memória.

Classe	Método	Format (getEncoded)
Chave	getEncoded()	RAW
ECPrivateChave	getEncoded()	PKCS#8
	getS()	N/D
RSAPrivateCrtKey	getEncoded()	X.509
	getPrivateExponent()	N/D
	getPrimeP()	N/D

Classe	Método	Format (getEncoded)
	getPrimeQ()	N/D
	getPrimeExponentP ()	N/D
	getPrimeExponentQ ()	N/D
	getCrtCoefficient()	N/D

Se você quiser restaurar o comportamento padrão e não permitir que o JCE exporte as chaves em branco, execute o seguinte comando:

Linux

```
$ /opt/cloudhsm/bin/configure-jce --disable-clear-key-extraction-in-software
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --disable-clear-key-extraction-in-software
```

Repita os comandos do JCE para AWS CloudHSM

AWS CloudHSM O SDK do cliente 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações controladas pelo HSM do lado do cliente. Quando um HSM acelera as operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, o cliente SDKs tentará repetir as operações limitadas até 3 vezes enquanto recua exponencialmente. Essa estratégia de repetição automática pode ser configurada para um dos dois modos: desativado e padrão.

- desativado: o Client SDK não executará nenhuma estratégia de repetição para nenhuma operação com controle de utilização pelo HSM.
- padrão: esse é o modo padrão para o Client SDK 5.8.0 e versões posteriores. Nesse modo, o cliente SDKs repetirá automaticamente as operações limitadas recuando exponencialmente.

Para obter mais informações, consulte [Controle de utilização do HSM](#).

Definir os comandos de repetição para o modo desativado

Linux

Para definir comandos de repetição off para o Client SDK 5 no Linux

- Use os comandos a seguir para gerenciar as configurações do modo off:

```
$ sudo /opt/cloudhsm/bin/configure-jce --default-retry-mode off
```

Windows

Para definir comandos de repetição off para o Client SDK 5 no Windows

- Use os comandos a seguir para gerenciar as configurações do modo off:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --default-retry-mode off
```

Usando a versão anterior do SDK para trabalhar com AWS CloudHSM

 As versões 5.8.0 e anteriores do SDK chegaram ao fim do suporte. Depois de 31 de março de 2025, a documentação para as versões 3.4.4 e anteriores do SDK não estará mais disponível.

AWS CloudHSM inclui duas versões principais do SDK do cliente:

- Client SDK 5: este é o nosso SDK de cliente padrão e mais recente. Para obter informações sobre os benefícios e vantagens que ela oferece, consulte [Benefícios do AWS CloudHSM Client SDK 5](#).

- Client SDK 3: este é o nosso SDK do cliente mais antigo. Ele inclui um conjunto completo de componentes para ferramentas de gerenciamento e compatibilidade de aplicativos baseados em plataformas e linguagens.

Para obter instruções sobre como migrar do Client SDK 3 para o Client SDK 5, consulte [Migração do SDK do AWS CloudHSM cliente 3 para o SDK do cliente 5](#).

Este tópico descreve o Client SDK 3. Para ver qual versão do Client SDK você está usando, consulte [Verifique sua versão AWS CloudHSM do SDK do cliente](#).

Para fazer download do [Downloads](#).

Tópicos

- [Atualize AWS CloudHSM o Client SDK 3 no Linux](#)
- [AWS CloudHSM Plataformas compatíveis com o Client SDK 3](#)
- [Biblioteca PKCS #11 para AWS CloudHSM Client SDK 3](#)
- [Mecanismo AWS CloudHSM dinâmico OpenSSL para cliente SDK 3](#)
- [Provedor de JCE para AWS CloudHSM Client SDK 3](#)
- [API de criptografia: próxima geração \(CNG\) e provedores de armazenamento de chaves \(KSP\) para AWS CloudHSM](#)

Atualize AWS CloudHSM o Client SDK 3 no Linux

 As versões 5.8.0 e anteriores do SDK chegaram ao fim do suporte. Depois de 31 de março de 2025, a documentação para as versões 3.4.4 e anteriores do SDK não estará mais disponível.

Com AWS CloudHSM o Client SDK 3.1 e versões posteriores, a versão do daemon do cliente e todos os componentes que você instalar devem corresponder à atualização. Para todos os sistemas baseados em Linux, você deve usar um único comando para atualizar em lote o daemon do cliente com a mesma versão da biblioteca PKCS #11, do provedor do Java Cryptographic Extension (JCE – Extensão de criptografia do Java) ou do OpenSSL Dynamic Engine. Esse requisito não se aplica a sistemas baseados no Windows porque os binários para a biblioteca do fornecedor KSP e CNG já estão incluídos no pacote daemon do cliente.

Como verificar a versão do daemon do cliente

- Em um sistema Linux baseado em Red Hat (incluindo Amazon Linux e CentOS), use o seguinte comando:

```
rpm -qa | grep ^cloudhsm
```

- Em um sistema Linux baseado em Debian, use o seguinte comando:

```
apt list --installed | grep ^cloudhsm
```

- Em um sistema Windows, use o seguinte comando:

```
wmic product get name,version
```

Tópicos

- [Pré-requisitos](#)
- [Etapa 1: interromper o daemon do cliente](#)
- [Etapa 2: atualizar o Client SDK](#)
- [Etapa 3: iniciar o daemon do cliente](#)

Pré-requisitos

Baixe a versão mais recente do daemon do AWS CloudHSM cliente e escolha seus componentes.

Note

Você não precisa instalar todas as bibliotecas. Para cada biblioteca instalada, é necessário atualizar essa biblioteca para corresponder à versão daemon do cliente.

Daemon de cliente Linux mais recente

Amazon Linux

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

RHEL 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

RHEL 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client_latest_u18.04_amd64.deb
```

Biblioteca PKCS #11 mais recente

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-pkcs11-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-pkcs11_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-pkcs11_latest_u18.04_amd64.deb
```

OpenSSL Dynamic Engine mais recente

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-dyn_latest_amd64.deb
```

Provedor JCE mais recente

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-jce-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-jce_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-jce_latest_u18.04_amd64.deb
```

Etapa 1: interromper o daemon do cliente

Use o comando a seguir para parar o daemon do cliente.

Amazon Linux

```
$ sudo stop cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

CentOS 8

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client stop
```


Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client stop
```

Etapa 2: atualizar o Client SDK

O comando a seguir mostra a sintaxe necessária para atualizar o daemon e os componentes do cliente. Antes de executar o comando, remova todas as bibliotecas que você não pretende atualizar.

Amazon Linux

```
$ sudo yum install ./cloudhsm-client-latest.el6.x86_64.rpm \  
    <./cloudhsm-client-pkcs11-latest.el6.x86_64.rpm> \  
    <./cloudhsm-client-dyn-latest.el6.x86_64.rpm> \  
    <./cloudhsm-client-jce-latest.el6.x86_64.rpm>
```

Amazon Linux 2

```
$ sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm \  
    <./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \  
    <./cloudhsm-client-dyn-latest.el7.x86_64.rpm> \  
    <./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

CentOS 7

```
$ sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm \  
    <./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \  
    <./cloudhsm-client-dyn-latest.el7.x86_64.rpm> \  
    <./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

CentOS 8

```
$ sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm \  
    <./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm> \  
    <./cloudhsm-client-jce-latest.el8.x86_64.rpm>
```

RHEL 7

```
$ sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm \  
    <./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \  
    <./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

```
<./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \  
<./cloudhsm-client-dyn-latest.el7.x86_64.rpm> \  
<./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

RHEL 8

```
$ sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm \  
<./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm> \  
<./cloudhsm-client-jce-latest.el8.x86_64.rpm>
```

Ubuntu 16.04 LTS

```
$ sudo apt install ./cloudhsm-client_latest_amd64.deb \  
<cloudhsm-client-pkcs11_latest_amd64.deb> \  
<cloudhsm-client-dyn_latest_amd64.deb> \  
<cloudhsm-client-jce_latest_amd64.deb>
```

Ubuntu 18.04 LTS

```
$ sudo apt install ./cloudhsm-client_latest_u18.04_amd64.deb \  
<cloudhsm-client-pkcs11_latest_amd64.deb> \  
<cloudhsm-client-jce_latest_amd64.deb>
```

Etapa 3: iniciar o daemon do cliente

Use os comandos a seguir para iniciar o daemon do cliente.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

CentOS 8

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 8

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 20.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

AWS CloudHSM Plataformas compatíveis com o Client SDK 3

 As versões 5.8.0 e anteriores do SDK chegaram ao fim do suporte. Depois de 31 de março de 2025, a documentação para as versões 3.4.4 e anteriores do SDK não estará mais disponível.

AWS CloudHSM O Client SDK 3 requer um daemon de cliente e oferece ferramentas de linha de comando, incluindo o CloudHSM Management Utility (CMU), o Key Management Utility (KMU) e a ferramenta de configuração.

O suporte básico é diferente para cada versão do SDK AWS CloudHSM do cliente. O suporte à plataforma para componentes em um SDK normalmente, mas nem sempre, corresponde ao suporte

básico. Para determinar o suporte à plataforma para um componente específico, primeiro certifique-se de que a plataforma desejada apareça na seção base do SDK e, em seguida, verifique se há exclusões ou qualquer outra informação pertinente na seção de componentes.

O suporte da plataforma muda com o tempo. Versões anteriores do Client SDK do CloudHSM podem não ser compatíveis com todos os sistemas operacionais listados aqui. Use as notas de versão para determinar o suporte do sistema operacional para versões anteriores do Client SDK do CloudHSM. Para obter mais informações, consulte [Downloads para o SDK AWS CloudHSM do cliente](#).

AWS CloudHSM suporta somente sistemas operacionais de 64 bits.

Tópicos

- [Suporte Linux para o AWS CloudHSM Client SDK 3](#)
- [Suporte do Windows para o AWS CloudHSM Client SDK 3](#)
- [Compatibilidade com HSM para o AWS CloudHSM Client SDK 3](#)

Suporte Linux para o AWS CloudHSM Client SDK 3

AWS CloudHSM O Client SDK 3 é compatível com os seguintes sistemas operacionais e plataformas Linux.

- Amazon Linux
- Amazon Linux 2
- CentOS 6.10+ ²
- CentOS 7.3+
- CentOS 8 ^{1,4}
- Red Hat Enterprise Linux (RHEL) 6.10+ ²
- Red Hat Enterprise Linux (RHEL) 7.3+
- Red Hat Enterprise Linux (RHEL) 8 ¹
- Ubuntu 16.04 LTS ³
- Ubuntu 18.04 LTS ¹

[1] Não há suporte para o OpenSSL Dynamic Engine. Para obter mais informações, consulte [OpenSSL Dynamic Engine](#).

[2] Não há suporte para o Client SDK 3.3.0 e versões posteriores.

[3] O SDK 3.4 é a última versão suportada no Ubuntu 16.04.

[4] O SDK 3.4 é a última versão suportada no CentOS 8.3+.

Suporte do Windows para o AWS CloudHSM Client SDK 3

AWS CloudHSM O Client SDK 3 oferece suporte às seguintes versões do Windows Server.

- Microsoft Windows Server 2012
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Compatibilidade com HSM para o AWS CloudHSM Client SDK 3

A tabela a seguir descreve a compatibilidade AWS CloudHSM do Client SDK 3 para HSMs.

hsm1.medium	hsm2m.medium
Compatível com o Client SDK versão 3.1.0 e posteriores.	Sem compatibilidade.

Biblioteca PKCS #11 para AWS CloudHSM Client SDK 3

O PKCS #11 é um padrão para realizar operações criptográficas em Módulos de segurança de hardware (HSM) no AWS CloudHSM.

Para obter informações sobre bootstrapping, consulte [Conectar-se ao cluster](#).

Tópicos

- [Instale a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3](#)
- [Autentique-se na biblioteca PKCS #11 do AWS CloudHSM Client SDK 3](#)
- [Tipos de chaves compatíveis com a biblioteca PKCS #11 para AWS CloudHSM Client SDK 3](#)
- [Mecanismos compatíveis com o AWS CloudHSM Client SDK 3](#)

- [Operações de API suportadas para o AWS CloudHSM Client SDK 3](#)
- [Atributos principais na biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3](#)
- [Exemplos de código para a biblioteca PKCS #11 para AWS CloudHSM Client SDK 3](#)

Instale a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3

Este tópico fornece instruções para instalar a biblioteca PKCS #11 para a série AWS CloudHSM Client SDK 3. Para obter mais informações sobre a biblioteca Client SDK ou do PKCS #11, consulte [Usando a biblioteca Client SDK](#) e [PKCS #11](#).

Pré-requisitos para o Client SDK 3

A biblioteca PKCS #11 requer o AWS CloudHSM cliente.

Se você não instalou e configurou o AWS CloudHSM cliente, faça isso agora seguindo as etapas em [Instalar o cliente \(Linux\)](#). Após instalar e configurar o cliente, use o comando a seguir para iniciá-lo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

CentOS 8

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 8

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 20.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Biblioteca PKCS #11 para o Client SDK 3

O comando a seguir faz download da biblioteca do PKCS #11 e a instala.

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-pkcs11-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-pkcs11_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-pkcs11_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-pkcs11_latest_u18.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-pkcs11_latest_u18.04_amd64.deb
```


- Se a instância EC2 na qual você instalou a biblioteca do PKCS #11 não tiver outros componentes do Client SDK 3 instalados, você deverá inicializar o Client SDK 3. Você só precisa fazer isso uma vez em cada instância com um componente do Client SDK 3.
- Você pode encontrar os arquivos da biblioteca do PKCS #11 nos seguintes locais:

Binários, scripts de configuração, certificados e arquivos de log do Linux:

```
/opt/cloudhsm/lib
```

Autentique-se na biblioteca PKCS #11 do AWS CloudHSM Client SDK 3

Quando você usa a biblioteca PKCS #11, seu aplicativo é executado como um [usuário criptográfico \(UC\)](#) específico em seu HSMs in. AWS CloudHSM O aplicativo pode visualizar e gerenciar apenas as chaves que o CU possui e compartilha. Você pode usar uma UC existente na sua UC HSMs ou criar uma nova UC. Para obter informações sobre gerenciamento CUs, consulte [Gerenciando usuários do HSM com o CloudHSM CLI e Gerenciando usuários do HSM com o CloudHSM Management Utility \(CMU\)](#).

Para especificar o CU para a biblioteca PKCS #11, use o parâmetro do PIN da [função C_Login](#) do PKCS #11. Para AWS CloudHSM, o parâmetro pin tem o seguinte formato:

```
<CU_user_name>:<password>
```

Por exemplo, o seguinte comando define o PIN de PKCS #11 como o CU com nome de usuário CryptoUser e senha CUPassword123!.

```
CryptoUser:CUPassword123!
```

Tipos de chaves compatíveis com a biblioteca PKCS #11 para AWS CloudHSM Client SDK 3

A biblioteca PKCS #11 oferece suporte aos seguintes tipos de chave com o AWS CloudHSM Client SDK 3.

Tipo de chave	Description
RSA	Gera chaves RSA de 2.048 a 4.096 bits, em incrementos de 256 bits.
EC	Gera chaves com as curvas secp224r1 (P-224), secp256r1 (P-256), secp256k1 (Blockchain), secp384r1 (P-384) e secp521r1 (P-521).
AES	Gere chaves AES de 128, 192 e 256 bits.
DES3 (DES triplo)	Gere chaves de 192 bits. DES3 Consulte a nota 1 abaixo para ver uma mudança futura.
GENERIC_SECRET	Gerar chaves genéricas de 1 a 64 bytes.

- [1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Mecanismos compatíveis com o AWS CloudHSM Client SDK 3

A biblioteca PKCS #11 oferece suporte aos seguintes algoritmos para o AWS CloudHSM Client SDK 3:

- Criptografia e descriptografia — AES-CBC, AES-CTR, AES-ECB, AES-GCM, -CBC, -ECB, RSA-OAEP e RSA-PKCS DES3 DES3
- Assinar e verificar: RSA, HMAC e ECDSA; com e sem hash
- Hash/digest — SHA1,, SHA224, e SHA256 SHA384 SHA512
- Encapsulamento de chave: AES Key Wrap⁴, AES-GCM, RSA-AES e RSA-OAEP
- Derivação de chave — ECDH, ⁵ SP800-108 CTR KDF

A tabela de funções do mecanismo da biblioteca PKCS #11

A biblioteca PKCS #11 é compatível com a versão 2.40 da especificação PKCS #11. Para invocar um recurso de criptografia usando o PKCS#11, chame uma função com um determinado mecanismo. A tabela a seguir resume as combinações de funções e mecanismos suportados pelo AWS CloudHSM.

Interpretar a tabela da função do mecanismo PKCS #11 compatível

Uma marca ✓ indica que AWS CloudHSM suporta o mecanismo da função. Não oferecemos suporte a todas as funções possíveis listadas na especificação do PKCS #11. Uma marca ✕ indica que ainda AWS CloudHSM não suporta o mecanismo para a função dada, mesmo que o padrão PKCS #11 o permita. Células vazias indicam que o PKCS #11 padrão não oferece suporte ao mecanismo para a função determinada.

Mecanismos e funções do PKCS #11 compatíveis

Mecanismo	Funções						
	Gerar chave ou par de chave	Assinar e verificar	SR e VR	Resumo	Criptogra far e descripto grafar	Derivar chave	Embrulhe e UnWrap
CKM_RSA_P KCS_KEY_P AIR_GEN	✓						
CKM_RSA_X 9_31_KEY_ PAIR_GEN	✓ ²						
CKM_RSA_X _509		✓			✓		
CKM_RSA_P KCS consulte a		✓ ¹	✕		✓ ¹		✓ ¹

Mecanismo	Funções						
observação							
o 8							
CKM_RSA_P KCS_OAEP					✓ 1		✓ 6
CKM_SHA1_ RSA_PKCS		✓ 3.2					
CKM_SHA22 4_RSA_PKC S		✓ 3.2					
CKM_SHA25 6_RSA_PKC S		✓ 3.2					
CKM_SHA38 4_RSA_PKC S		✓ 2,3.2					
CKM_SHA51 2_RSA_PKC S		✓ 3.2					
CKM_RSA_P KCS_PSS		✓ 1					
CKM_SHA1_ RSA_PKCS_ PSS		✓ 3.2					
CKM_SHA22 4_RSA_PKC S_PSS		✓ 3.2					

Mecanismo	Funções						
CKM_SHA256_RSA_PKCS_PSS		✓ 3.2					
CKM_SHA384_RSA_PKCS_PSS		✓ 2,3.2					
CKM_SHA512_RSA_PKCS_PSS		✓ 3.2					
CKM_EC_KEY_PAIR_GENERATION	✓						
CKM_ECDSA		✓ 1					
CKM_ECDSA_SHA1		✓ 3.2					
CKM_ECDSA_SHA224		✓ 3.2					
CKM_ECDSA_SHA256		✓ 3.2					
CKM_ECDSA_SHA384		✓ 3.2					
CKM_ECDSA_SHA512		✓ 3.2					
CKM_ECDH1_DERIVE						✓ 5	

Mecanismo	Funções						
CKM_SP800_108_COUNTER_KDF						✓	
CKM_GENERIC_SECRET_KEY_GEN	✓						
CKM_AES_KEY_GEN	✓						
CKM_AES_ECB					✓		✗
CKM_AES_CTR					✓		✗
CKM_AES_CBC					✓ 3.3		✗
CKM_AES_CBC_PAD					✓		✗
CKM_DES3_KEY_GEN consulte a observação o 8	✓						
CKM_DES3_CBC consulte a observação o 8					✓ 3.3		✗

Mecanismo	Funções						
CKM_DES3_					✓		✗
CBC_PAD							
consulte a							
observaçã							
o 8							
CKM_DES3_					✓		✗
ECB							
consulte a							
observaçã							
o 8							
CKM_AES_G					✓ 3.3 , 4		✓ 7.1
CM							
CKM_CLOUD					✓ 7.1		✓ 7.1
HSM_AES_G							
CM							
CKM_SHA_1				✓ 3.1			
CKM_SHA_1		✓ 3.3					
_HMAC							
CKM_SHA22				✓ 3.1			
4							
CKM_SHA22		✓ 3.3					
4_HMAC							
CKM_SHA25				✓ 3.1			
6							

Mecanismo	Funções						
CKM_SHA256_HMAC		✓ 3.3					
CKM_SHA384				✓ 3.1			
CKM_SHA384_HMAC		✓ 3.3					
CKM_SHA512				✓ 3.1			
CKM_SHA512_HMAC		✓ 3.3					
CKM_RSA_AES_KEY_WRAP							✓
CKM_AES_KEY_WRAP							✓
CKM_AES_KEY_WRAP_PAD							✓
CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD							✓ 7.1
CKM_CLOUDHSM_AES_KEY_WRAP_PAD_KCS5_PAD							✓ 7.1

Mecanismo	Funções						
CKM_CLOUD HSM_AES_K EY_WRAP_Z ERO_PAD							✓ 7.1

Anotações do mecanismo

- [1] somente operações de uma única parte.
- [2] O mecanismo é funcionalmente idêntico ao mecanismo CKM_RSA_PKCS_KEY_PAIR_GEN, mas oferece garantias mais fortes para a geração p e q.
- [3.1] AWS CloudHSM aborda o hashing de forma diferente com base no SDK do cliente. Para o Client SDK 3, onde fazemos o hashing depende do tamanho dos dados e se você está usando operações de uma ou várias partes.

Operações em uma única parte no Client SDK 3

A tabela 3.1 lista o tamanho máximo do conjunto de dados para cada mecanismo do Client SDK 3. O hash inteiro é computado dentro do HSM. Não há suporte para tamanhos de dados maiores que 16 KB.

Tabela 3.1: tamanho máximo do conjunto de dados para operações em uma única parte

Mecanismo	Tamanho máximo de dados
CKM_SHA_1	16296
CKM_SHA224	16264
CKM_SHA256	16296
CKM_SHA384	16232
CKM_SHA512	16232

Client SDK 3 de operações em várias partes

Suporta tamanhos de dados superiores a 16 KB, mas o tamanho dos dados determina onde o hash ocorre. Buffers de dados com menos de 16 KB são codificados dentro do HSM. Buffers entre 16 KB e o tamanho máximo de dados do seu sistema são codificados localmente no software. Lembre-se: as funções de hash não exigem segredos criptográficos, então é possível computá-las com segurança fora do HSM.

- [3.2] AWS CloudHSM aborda o hashing de forma diferente com base no SDK do cliente. Para o Client SDK 3, onde fazemos o hashing depende do tamanho dos dados e se você está usando operações de uma ou várias partes.

Client SDK 3 de operações em uma única parte

A tabela 3.2 lista o tamanho máximo do conjunto de dados para cada mecanismo do Client SDK 3. Não há suporte para tamanhos de dados maiores que 16 KB.

Tabela 3.2: Tamanho máximo do conjunto de dados para operações em uma única parte

Mecanismo	Tamanho máximo de dados
CKM_SHA1_RSA_PKCS	16296
CKM_SHA224_RSA_PKCS	16264
CKM_SHA256_RSA_PKCS	16296
CKM_SHA384_RSA_PKCS	16232
CKM_SHA512_RSA_PKCS	16232
CKM_SHA1_RSA_PKCS_PSS	16296
CKM_SHA224_RSA_PKCS_PSS	16264
CKM_SHA256_RSA_PKCS_PSS	16296
CKM_SHA384_RSA_PKCS_PSS	16232
CKM_SHA512_RSA_PKCS_PSS	16232
CKM_ECDSA_SHA1	16296

Mecanismo	Tamanho máximo de dados
CKM_ECDSA_SHA224	16264
CKM_ECDSA_SHA256	16296
CKM_ECDSA_SHA384	16232
CKM_ECDSA_SHA512	16232

Client SDK 3 de operações em várias partes

Suporta tamanhos de dados superiores a 16 KB, mas o tamanho dos dados determina onde o hash ocorre. Buffers de dados com menos de 16 KB são codificados dentro do HSM. Buffers entre 16 KB e o tamanho máximo de dados do seu sistema são codificados localmente no software. Lembre-se: as funções de hash não exigem segredos criptográficos, então é possível computá-las com segurança fora do HSM.

- [3.3] Ao trabalhar com dados usando qualquer um dos mecanismos a seguir, se o buffer de dados exceder o tamanho máximo de dados, a operação resultará em um erro. Para esses mecanismos, todo o processamento de dados deve ocorrer dentro do HSM. A tabela a seguir lista o tamanho máximo de dados definido para cada mecanismo:

Tabela 3.3: Tamanho máximo do conjunto de dados

Mecanismo	Tamanho máximo de dados
CKM_SHA_1_HMAC	16288
CKM_SHA224_HMAC	16256
CKM_SHA256_HMAC	16288
CKM_SHA384_HMAC	16224
CKM_SHA512_HMAC	16224
CKM_AES_CBC	16272
CKM_AES_GCM	16224

Mecanismo	Tamanho máximo de dados
CKM_CLOUDHSM_AES_GCM	16224
CKM_DES3_CBC	16280

- [4] Ao executar a criptografia AES-GCM, o HSM não aceitará dados do vetor de inicialização (IV) do aplicativo. Você deve usar um IV gerado por ele. O IV de 12 bytes fornecido pelo HSM é gravado na referência da memória apontada pelo elemento pIV da estrutura de parâmetros CK_GCM_PARAMS que você fornece. Para que não haja confusão para o usuário, o SDK do PKCS #11 na versão 1.1.1 e posterior, garante que o pIV aponte para um buffer zerado quando a criptografia AES-GCM é inicializada.
- [5] Somente o Client SDK 3. O mecanismo está implementado para oferecer suporte a casos de descarregamento de SSL/TLS e é executado somente parcialmente no HSM. Antes de usar esse mecanismo, consulte "Problema: a derivação de chaves ECDH é executada apenas parcialmente no HSM" em [Problemas conhecidos da biblioteca PKCS #11 para AWS CloudHSM](#). CKM_ECDH1_DERIVE não é compatível com a curva secp521r1 (P-521).
- [6] Os seguintes CK_MECHANISM_TYPE e CK_RSA_PKCS_MGF_TYPE são compatíveis como CK_RSA_PKCS_OAEP_PARAMS para CKM_RSA_PKCS_OAEP:
 - CKM_SHA_1 usando CKG_MGF1_SHA1
 - CKM_SHA224 usando CKG_MGF1_SHA224
 - CKM_SHA256 usando CKG_MGF1_SHA256
 - CKM_SHA384 usando CKM_MGF1_SHA384
 - CKM_SHA512 usando CKM_MGF1_SHA512
- [7.1] Mecanismo definido pelo fornecedor. Para usar os mecanismos definidos pelo fornecedor do CloudHSM, os aplicativos PKCS #11 devem incluir /opt/cloudhsm/include/pkcs11t.h durante a compilação.

CKM_CLOUDHSM_AES_GCM: Este mecanismo proprietário é uma alternativa programaticamente mais segura para o padrão CKM_AES_GCM. Ele antecede o IV gerado pelo HSM para o texto cifrado em vez de escrevê-lo de volta na estrutura CK_GCM_PARAMS fornecida durante a inicialização da cifra. Você pode usar esse mecanismo com as funções C_Encrypt, C_WrapKey, C_Decrypt, e C_UnwrapKey. Ao usar esse mecanismo, a variável pIV no struct CK_GCM_PARAMS deve ser definida como NULL. Ao usar este mecanismo com C_Decrypt e C_UnwrapKey, espera-se que o IV seja precedido pelo texto cifrado que está sendo desencapsulado.

CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD: Encapsulamento de chaves AES com preenchimento PKCS #5

CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD: Encapsulamento de chaves AES com preenchimento de zeros

Para obter opções adicionais de empacotamento de chaves AES, consulte [Empacotamento de chaves AES](#).

- [8] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Operações de API suportadas para o AWS CloudHSM Client SDK 3

A biblioteca PKCS #11 oferece suporte às seguintes operações de API PKCS #11 para o AWS CloudHSM Client SDK 3.

- C_CloseAllSessions
- C_CloseSession
- C_CreateObject
- C_Decrypt
- C_DecryptFinal
- C_DecryptInit
- C_DecryptUpdate
- C_DeriveKey
- C_DestroyObject
- C_Digest
- C_DigestFinal
- C_DigestInit
- C_DigestUpdate
- C_Encrypt
- C_EncryptFinal
- C_EncryptInit

- C_EncryptUpdate
- C_Finalize
- C_FindObjects
- C_FindObjectsFinal
- C_FindObjectsInit
- C_GenerateKey
- C_GenerateKeyPair
- C_GenerateRandom
- C_GetAttributeValue
- C_GetFunctionList
- C_GetInfo
- C_GetMechanismInfo
- C_GetMechanismList
- C_GetSessionInfo
- C_GetSlotInfo
- C_GetSlotList
- C_GetTokenInfo
- C_Initialize
- C_Login
- C_Logout
- C_OpenSession
- C_Sign
- C_SignFinal
- C_SignInit
- C_SignRecover (Somente suporte ao Client SDK 3)
- C_SignRecoverInit (Somente suporte ao Client SDK 3)
- C_SignUpdate
- C_UnWrapKey
- C_Verify
- C_VerifyFinal

- C_VerifyInit
- C_VerifyRecover (Somente suporte ao Client SDK 3)
- C_VerifyRecoverInit (Somente suporte ao Client SDK 3)
- C_VerifyUpdate
- C_WrapKey

Atributos principais na biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3

Um objeto pode ser uma chave pública, privada ou secreta. As ações permitidas em um objeto de chave são especificadas por meio de atributos. Os atributos são definidos quando o objeto de chave é criado. Quando você usa a biblioteca PKCS #11 para AWS CloudHSM, atribuímos valores padrão conforme especificado pelo padrão PKCS #11.

AWS CloudHSM não suporta todos os atributos listados na especificação PKCS #11. Estamos em conformidade com a especificação de todos os atributos aos quais oferecemos suporte. Esses atributos estão listados nas respectivas tabelas.

Funções criptográficas como C_CreateObject, C_GenerateKey, C_GenerateKeyPair, C_UnwrapKey, e C_DeriveKey que criam, modificam ou copiam objetos usam um modelo de atributo como um de seus parâmetros. Para obter mais informações sobre como passar um modelo de atributo durante a criação do objeto, consulte o exemplo para [Gerar chaves por meio da biblioteca PKCS #11](#).

Os tópicos a seguir fornecem mais informações sobre os AWS CloudHSM principais atributos do Client SDK 3.

Tópicos

- [Tabela de atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3](#)
- [Modificando os atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3](#)
- [Interpretação dos códigos de erro da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3](#)

Tabela de atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3

A tabela da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3 contém uma lista de atributos que diferem por tipo de chave. Ele indica se um determinado atributo é compatível com um determinado tipo de chave ao usar uma função criptográfica específica com AWS CloudHSM.

Legenda:

- ✓ indica que o CloudHSM oferece suporte ao atributo para o tipo de chave específico.
- ✗ indica que o CloudHSM não oferece suporte ao atributo para o tipo de chave específico.
- R indica que o valor do atributo é definido como somente leitura para o tipo de chave específico.
- S indica que o atributo não pode ser lido pelo `GetAttributeValue` pois ele é confidencial.
- Uma célula vazia na coluna Valor padrão indica que não há nenhum valor padrão específico atribuído ao atributo.

GenerateKeyPair

Atributo	Tipo de chave				Valor padrão
	EC privada	EC pública	RSA privada	RSA pública	
CKA_CLASS	✓	✓	✓	✓	
CKA_KEY_TYPE	✓	✓	✓	✓	
CKA_LABEL	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	Verdadeiro
CKA_TOKEN	✓	✓	✓	✓	Falso
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro

Atributo	Tipo de chave				Valor padrão
CKA_ENCRYPT	×	✓	×	✓	Falso
CKA_DECRYPT	✓	×	✓	×	Falso
CKA_DERIVE	✓	✓	✓	✓	Falso
CKA_MODIFIABLE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	×	✓	×	Falso
CKA_SIGN_RECOVER	×	×	✓ ³	×	
CKA_VERIFY	×	✓	×	✓	Falso
CKA_VERIFY_RECOVER	×	×	×	✓ ⁴	
CKA_WRAP	×	✓	×	✓	Falso
CKA_WRAP_TEMPLATE	×	✓	×	✓	
CKA_TRUSTED	×	✓	×	✓	Falso

Atributo	Tipo de chave				Valor padrão
CKA_WRAP_WITH_TRUSTED	✓	✗	✓	✗	Falso
CKA_UNWRAP	✓	✗	✓	✗	Falso
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	
CKA_SENSITIVE	✓	✗	✓	✗	Verdadeiro
CKA_ALWAYS_SENSITIVE	R	✗	R	✗	
CKA_EXTRACTABLE	✓	✗	✓	✗	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	✗	R	✗	
CKA_MODULUS	✗	✗	✗	✗	
CKA_MODULUS_BITS	✗	✗	✗	✓ ²	
CKA_PRIME_1	✗	✗	✗	✗	
CKA_PRIME_2	✗	✗	✗	✗	

Atributo	Tipo de chave					Valor padrão
CKA_COEFFICIENT		×	×	×	×	
CKA_EXPONENT_1		×	×	×	×	
CKA_EXPONENT_2		×	×	×	×	
CKA_PRIVATE_EXPONENT		×	×	×	×	
CKA_PUBLIC_EXPONENT		×	×	×	✓ ²	
CKA_EC_PARAMS		×	✓ ²	×	×	
CKA_EC_POINT		×	×	×	×	
CKA_VALUE		×	×	×	×	
CKA_VALUE_LEN		×	×	×	×	
CKA_CHECK_VALUE		R	R	R	R	

GenerateKey

Atributo	Tipo de chave			Valor padrão
	AES	DES3	Segredo genérico	
CKA_CLASS	✓	✓	✓	
CKA_KEY_TYPE	✓	✓	✓	
CKA_LABEL	✓	✓	✓	
CKA_ID	✓	✓	✓	
CKA_LOCAL	R	R	R	Verdadeiro
CKA_TOKEN	✓	✓	✓	Falso
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT	✓	✓	✗	Falso
CKA_DECRYPT	✓	✓	✗	Falso
CKA_DERIVE	✓	✓	✓	Falso
CKA_MODIFIABLE	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✓	✓	Verdadeiro

Atributo	Tipo de chave			Valor padrão
CKA_SIGN_RECOVER	✗	✗	✗	
CKA_VERIFY	✓	✓	✓	Verdadeiro
CKA_VERIFY_RECOVER	✗	✗	✗	
CKA_WRAP	✓	✓	✗	Falso
CKA_WRAP_TEMPLATE	✓	✓	✗	
CKA_TRUSTED	✓	✓	✗	Falso
CKA_WRAP_WITH_TRUSTED	✓	✓	✓	Falso
CKA_UNWRAP	✓	✓	✗	Falso
CKA_UNWRAP_TEMPLATE	✓	✓	✗	
CKA_SENSITIVE	✓	✓	✓	Verdadeiro
CKA_ALWAYS_SENSITIVE	✗	✗	✗	

Atributo	Tipo de chave				Valor padrão
CKA_EXTRACTABLE		✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE		R	R	R	
CKA_MODULUS		✗	✗	✗	
CKA_MODULUS_BITS		✗	✗	✗	
CKA_PRIME_1		✗	✗	✗	
CKA_PRIME_2		✗	✗	✗	
CKA_COEFFICIENT		✗	✗	✗	
CKA_EXPONENT_1		✗	✗	✗	
CKA_EXPONENT_2		✗	✗	✗	
CKA_PRIVATE_EXPONENT		✗	✗	✗	
CKA_PUBLIC_EXPONENT		✗	✗	✗	

Atributo	Tipo de chave			Valor padrão
CKA_EC_PA RAMS	×	×	×	
CKA_EC_PO INT	×	×	×	
CKA_VALUE	×	×	×	
CKA_VALUE _LEN	✓ ²	×	✓ ²	
CKA_CHECK _VALUE	R	R	R	

CreateObject

Atributo	Tipo de chave							Valor padrão
	EC privada	EC pública	RSA privada	RSA pública	AES	DES3	Segredo genérico	
CKA_CLASS	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	
CKA_KEY_T YPE	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²	
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	R	R	R	Falso

Atributo	Tipo de chave							Valor padrão
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓	Falso
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT	✗	✗	✗	✓	✓	✓	✗	Falso
CKA_DECRYPT	✗	✗	✓	✗	✓	✓	✗	Falso
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	Falso
CKA_MODIFIABLE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✗	✓	✗	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✗	✗	✓ ³	✗	✗	✗	✗	Falso
CKA_VERIFY	✗	✓	✗	✓	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✗	✗	✗	✓ ⁴	✗	✗	✗	
CKA_WRAP	✗	✗	✗	✓	✓	✓	✗	Falso
CKA_WRAP_TEMPLATE	✗	✓	✗	✓	✓	✓	✗	

Atributo	Tipo de chave							Valor padrão
CKA_TRUSTED	✗	✓	✗	✓	✓	✓	✗	Falso
CKA_WRAP_WITH_TRUSTED	✓	✗	✓	✗	✓	✓	✓	Falso
CKA_UNWRAP	✗	✗	✓	✗	✓	✓	✗	Falso
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	✓	✓	✗	
CKA_SENSITIVE	✓	✗	✓	✗	✓	✓	✓	Verdadeiro
CKA_ALWAYS_SENSITIVE	R	✗	R	✗	R	R	R	
CKA_EXTRACTABLE	✓	✗	✓	✗	✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	✗	R	✗	R	R	R	
CKA_MODULUS	✗	✗	✓ ²	✓ ²	✗	✗	✗	
CKA_MODULUS_BITS	✗	✗	✗	✗	✗	✗	✗	
CKA_PRIME_1	✗	✗	✓	✗	✗	✗	✗	

Atributo	Tipo de chave							Valor padrão
CKA_PRIME_2	×	×	✓	×	×	×	×	
CKA_COEFFICIENT	×	×	✓	×	×	×	×	
CKA_EXPONENT_1	×	×	✓	×	×	×	×	
CKA_EXPONENT_2	×	×	✓	×	×	×	×	
CKA_PRIVATE_EXPONENT	×	×	✓ ²	×	×	×	×	
CKA_PUBLIC_EXPONENT	×	×	✓ ²	✓ ²	×	×	×	
CKA_EC_PARAMS	✓ ²	✓ ²	×	×	×	×	×	
CKA_EC_POINT	×	✓ ²	×	×	×	×	×	
CKA_VALUE	✓ ²	×	×	×	✓ ²	✓ ²	✓ ²	
CKA_VALUE_LEN	×	×	×	×	×	×	×	
CKA_CHECK_VALUE	R	R	R	R	R	R	R	

UnwrapKey

Atributo	Tipo de chave						Valor padrão
	EC privada	RSA privada	AES	DES3	Segredo genérico		
CKA_CLASS	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²		
CKA_KEY_TYPE	✓ ²	✓ ²	✓ ²	✓ ²	✓ ²		
CKA_LABEL	✓	✓	✓	✓	✓		
CKA_ID	✓	✓	✓	✓	✓		
CKA_LOCAL	R	R	R	R	R		Falso
CKA_TOKEN	✓	✓	✓	✓	✓		Falso
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹		Verdadeiro
CKA_ENCRYPT	✗	✗	✓	✓	✗		Falso
CKA_DECRYPT	✗	✓	✓	✓	✗		Falso
CKA_DERIVE	✓	✓	✓	✓	✓		Falso
CKA_MODIFIABLE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹		Verdadeiro

Atributo	Tipo de chave						Valor padrão
CKA_DESTR OYABLE		✓	✓	✓	✓	✓	Verdadeir o
CKA_SIGN		✓	✓	✓	✓	✓	Falso
CKA_SIGN_ RECOVER		✗	✓ ³	✗	✗	✗	Falso
CKA_VERIF Y		✗	✗	✓	✓	✓	Falso
CKA_VERIF Y_RECOVER		✗	✗	✗	✗	✗	
CKA_WRAP		✗	✗	✓	✓	✗	Falso
CKA_UNWRA P		✗	✓	✓	✓	✗	Falso
CKA_SENSI TIVE		✓	✓	✓	✓	✓	Verdadeir o
CKA_EXTRA CTABLE		✓	✓	✓	✓	✓	Verdadeir o
CKA_NEVER _EXTRACTA BLE		R	R	R	R	R	
CKA_ALWAYS SENSITI VE		R	R	R	R	R	
CKA_MODUL US		✗	✗	✗	✗	✗	

Atributo	Tipo de chave						Valor padrão
CKA_MODULUS_BITS		×	×	×	×	×	
CKA_PRIME_1		×	×	×	×	×	
CKA_PRIME_2		×	×	×	×	×	
CKA_COEFFICIENT		×	×	×	×	×	
CKA_EXPONENT_1		×	×	×	×	×	
CKA_EXPONENT_2		×	×	×	×	×	
CKA_PRIVATE_EXPONENT		×	×	×	×	×	
CKA_PUBLIC_EXPONENT		×	×	×	×	×	
CKA_EC_PARAMS		×	×	×	×	×	
CKA_EC_POINT		×	×	×	×	×	
CKA_VALUE		×	×	×	×	×	

Atributo	Tipo de chave						Valor padrão
CKA_VALUE_LEN		×	×	×	×	×	
CKA_CHECK_VALUE		R	R	R	R	R	

DeriveKey

Atributo	Tipo de chave				Valor padrão
		AES	DES3	Segredo genérico	
CKA_CLASS		✓ ²	✓ ²	✓ ²	
CKA_KEY_TYPE		✓ ²	✓ ²	✓ ²	
CKA_LABEL		✓	✓	✓	
CKA_ID		✓	✓	✓	
CKA_LOCAL		R	R	R	Verdadeiro
CKA_TOKEN		✓	✓	✓	Falso
CKA_PRIVATE		✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_ENCRYPT		✓	✓	×	Falso
CKA_DECRYPT		✓	✓	×	Falso

Atributo	Tipo de chave			Valor padrão
CKA_DERIVE	✓	✓	✓	Falso
CKA_MODIFIABLE	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_DESTROYABLE	✓ ¹	✓ ¹	✓ ¹	Verdadeiro
CKA_SIGN	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✗	✗	✗	
CKA_VERIFY	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✗	✗	✗	
CKA_WRAP	✓	✓	✗	Falso
CKA_UNWRAP	✓	✓	✗	Falso
CKA_SENSITIVE	✓	✓	✓	Verdadeiro
CKA_EXTRACTABLE	✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	R	R	

Atributo	Tipo de chave			Valor padrão
CKA_ALWAYS_SENSITIVE	R	R	R	
CKA_MODULUS	×	×	×	
CKA_MODULUS_BITS	×	×	×	
CKA_PRIME_1	×	×	×	
CKA_PRIME_2	×	×	×	
CKA_COEFFICIENT	×	×	×	
CKA_EXPONENT_1	×	×	×	
CKA_EXPONENT_2	×	×	×	
CKA_PRIVATE_EXPONENT	×	×	×	
CKA_PUBLIC_EXPONENT	×	×	×	
CKA_EC_PARAMS	×	×	×	

Atributo	Tipo de chave			Valor padrão
CKA_EC_POINT		✗	✗	✗
CKA_VALUE		✗	✗	✗
CKA_VALUE_LEN		✓ ²	✗	✓ ²
CKA_CHECK_VALUE		R	R	R

GetAttributeValue

Atributo	Tipo de chave						
	EC privada	EC pública	RSA privada	RSA pública	AES	DES3	Segredo genérico
CKA_CLASS	✓	✓	✓	✓	✓	✓	✓
CKA_KEY_TYPE	✓	✓	✓	✓	✓	✓	✓
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓
CKA_ID	✓	✓	✓	✓	✓	✓	✓
CKA_LOCAL	✓	✓	✓	✓	✓	✓	✓
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓

Atributo	Tipo de chave						
CKA_PRIVATE	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹	✓ ¹
CKA_ENCRYPT	✗	✗	✗	✓	✓	✓	✗
CKA_DECRYPT	✗	✗	✓	✗	✓	✓	✗
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✗	✓	✗	✓	✓	✓
CKA_SIGN_RECOVER	✗	✗	✓	✗	✗	✗	✗
CKA_VERIFY	✗	✓	✗	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✗	✗	✗	✓	✗	✗	✗
CKA_WRAP	✗	✗	✗	✓	✓	✓	✗
CKA_WRAP_TEMPLATE	✗	✓	✗	✓	✓	✓	✗
CKA_TRUSTED	✗	✓	✗	✓	✓	✓	✓

Atributo	Tipo de chave						
CKA_WRAP_WITH_TRUSTED	✓	✗	✓	✗	✓	✓	✓
CKA_UNWRAP	✗	✗	✓	✗	✓	✓	✗
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	✓	✓	✗
CKA_SENSITIVE	✓	✗	✓	✗	✓	✓	✓
CKA_EXTRACTABLE	✓	✗	✓	✗	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✗	✓	✗	✓	✓	✓
CKA_ALWAYS_SENSITIVE	R	R;	R	R	R	R	R
CKA_MODULUS	✗	✗	✓	✓	✗	✗	✗
CKA_MODULUS_BITS	✗	✗	✗	✓	✗	✗	✗
CKA_PRIME_1	✗	✗	S	✗	✗	✗	✗
CKA_PRIME_2	✗	✗	S	✗	✗	✗	✗

Atributo	Tipo de chave						
CKA_COEFFICIENT	×	×	S	×	×	×	×
CKA_EXPONENT_1	×	×	S	×	×	×	×
CKA_EXPONENT_2	×	×	S	×	×	×	×
CKA_PRIVATE_EXPONENT	×	×	S	×	×	×	×
CKA_PUBLIC_EXPONENT	×	×	✓	✓	×	×	×
CKA_EC_PARAMS	✓	✓	×	×	×	×	×
CKA_EC_POINT	×	✓	×	×	×	×	×
CKA_VALUE	S	×	×	×	✓ ²	✓ ²	✓ ²
CKA_VALUE_LEN	×	×	×	×	✓	×	✓
CKA_CHECK_VALUE	✓	✓	✓	✓	✓	✓	×

Anotações de atributos

- [1] Este atributo tem suporte parcial do firmware e deve ser explicitamente definido apenas como o valor padrão.

- [2] Atributo obrigatório.
- [3] Somente o Client SDK 3. O atributo CKA_SIGN_RECOVER é derivado do atributo CKA_SIGN. Se estiver sendo definido, ele só poderá ser definido como o mesmo valor definido para CKA_SIGN. Se não estiver definido, ele derivará o valor padrão de CKA_SIGN. Como o CloudHSM só oferece suporte aos mecanismos de assinatura recuperáveis com base no RSA, esse atributo é aplicável somente às chaves públicas RSA no momento.
- [4] Somente o Client SDK 3. O atributo CKA_VERIFY_RECOVER é derivado do atributo CKA_VERIFY. Se estiver sendo definido, ele só poderá ser definido como o mesmo valor definido para CKA_VERIFY. Se não estiver definido, ele derivará o valor padrão de CKA_VERIFY. Como o CloudHSM só oferece suporte aos mecanismos de assinatura recuperáveis com base no RSA, esse atributo é aplicável somente às chaves públicas RSA no momento.

Modificando os atributos da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3

Alguns atributos de um objeto podem ser modificados depois que o objeto foi criado, enquanto alguns não podem. Para modificar atributos, use o comando [setAttribute](#) do `cloudhsm_mgmt_util`. Você também pode derivar uma lista de atributos e as constantes que os representam usando o comando [listAttribute](#) command de `cloudhsm_mgmt_util`.

A lista a seguir exibe os atributos cuja modificação é permitida após a criação do objeto:

- CKA_LABEL
- CKA_TOKEN

Note

A modificação é permitida somente para alterar uma chave de sessão para uma chave de token. Use o comando [listAttribute](#) command de `cloudhsm_mgmt_util` para alterar o valor do atributo.

- CKA_ENCRYPT
- CKA_DECRYPT
- CKA_SIGN
- CKA_VERIFY
- CKA_WRAP
- CKA_UNWRAP

- CKA_LABEL
- CKA_SENSITIVE
- CKA_DERIVE

 Note

Esse atributo oferece suporte à derivação de chaves. Ele deve ser `False` para todas as chaves públicas e não pode ser definido como `True`. Para chaves privadas EC e secretas, ele pode ser definido como `True` ou `False`.

- CKA_TRUSTED

 Note

Esse atributo pode ser definido como `True` ou `False` somente pelo Responsável pela criptografia (CO)

- CKA_WRAP_WITH_TRUSTED

 Note

Aplique esse atributo a uma chave de dados exportável para especificar que você só pode agrupar essa chave com chaves marcadas como `CKA_TRUSTED`. Depois que `CKA_WRAP_WITH_TRUSTED` for definido como verdadeiro, o atributo se torna somente para leitura e não é possível alterar ou remover o atributo.

Interpretação dos códigos de erro da biblioteca PKCS #11 para o AWS CloudHSM Client SDK 3

Especificar no modelo um atributo da biblioteca PKCS #11 que não tenha suporte de uma chave específica resultará em um erro. A tabela a seguir contém códigos de erro que são gerados quando as especificações são violadas:

Código de erro	Descrição
CKR_TEMPLATE_INCONSISTENT	Você receberá esse erro quando especificar um atributo no modelo de atributo, em que o atributo está em conformidade com a especific

Código de erro	Descrição
	ação do PKCS #11, mas não tem suporte do CloudHSM.
CKR_ATTRIBUTE_TYPE_INVALID	Você receberá esse erro quando recuperar o valor de um atributo, que está em conformidade com a especificação do PKCS #11, mas não tem suporte do CloudHSM.
CKR_ATTRIBUTE_INCOMPLETE	Você receberá esse erro quando não especificar o atributo obrigatório no modelo de atributo.
CKR_ATTRIBUTE_READ_ONLY	Você receberá esse erro quando especificar um atributo somente leitura no modelo de atributo.

Exemplos de código para a biblioteca PKCS #11 para AWS CloudHSM Client SDK 3

Os exemplos de código mostrados GitHub mostram como realizar tarefas básicas usando a biblioteca PKCS #11 para AWS CloudHSM.

Pré-requisitos do código de exemplo

Antes de executar os exemplos, realize as seguintes etapas para configurar o ambiente:

- Instale e configure a [biblioteca PKCS #11](#) para o Client SDK 3.
- Configure um [usuário de criptografia \(CU\)](#). Seu aplicativo usa essa conta do HSM para executar as amostras de código no HSM.

Exemplos de código

Exemplos de código para a biblioteca AWS CloudHSM de software para PKCS #11 estão disponíveis em [GitHub](#). Este repositório inclui exemplos de como fazer operações comuns usando PKCS #11, incluindo criptografia, descriptografia, assinatura e verificação.

- [Gerar chaves \(AES, RSA, EC\)](#)
- [Listar atributos de chave](#)

- [Criptografar e descriptografar dados com AES-GCM](#)
- [Encrypt and decrypt data with AES_CTR](#)
- [Encrypt and decrypt data with 3DES](#)
- [Assinar e verificar dados com RSA](#)
- [Derive keys using HMAC KDF](#)
- [Wrap and unwrap keys with AES using PKCS #5 padding](#)
- [Wrap and unwrap keys with AES using no padding](#)
- [Wrap and unwrap keys with AES using zero padding](#)
- [Wrap and unwrap keys with AES-GCM](#)
- [Encapsular e desencapsular chaves com RSA](#)

Mecanismo AWS CloudHSM dinâmico OpenSSL para cliente SDK 3

O AWS CloudHSM OpenSSL Dynamic Engine permite que você descarregue operações criptográficas em seu cluster CloudHSM por meio da API OpenSSL.

AWS CloudHSM O SDK 3 do cliente exige que um daemon do cliente se conecte ao cluster. Ele oferece suporte a:

- Geração de chave de RSA para chaves de 2048, 3072 e 4096 bits.
- Assinar/verificar RSA.
- Criptografar/descriptografar RSA.
- Geração de número aleatório que é criptograficamente seguro e validado para FIPS.

Use as seções a seguir para instalar e configurar o mecanismo AWS CloudHSM dinâmico do OpenSSL.

Tópicos

- [Pré-requisitos para o OpenSSL Dynamic Engine com Client SDK 3 AWS CloudHSM](#)
- [Instale o OpenSSL Dynamic Engine AWS CloudHSM para Client SDK 3](#)
- [Use o OpenSSL Dynamic Engine AWS CloudHSM para Client SDK 3](#)

Pré-requisitos para o OpenSSL Dynamic Engine com Client SDK 3 AWS CloudHSM

Para obter mais informações sobre as suporte de plataforma, consulte [AWS CloudHSM Plataformas compatíveis com o Client SDK 3](#).

Antes de usar o mecanismo AWS CloudHSM dinâmico para OpenSSL com o Client SDK 3, você precisa do cliente. AWS CloudHSM

O cliente é um daemon que estabelece comunicação end-to-end criptografada com o HSMs em seu cluster, e o mecanismo OpenSSL se comunica localmente com o cliente. Para instalar e configurar o AWS CloudHSM cliente, consulte [Instalar o cliente \(Linux\)](#). Use o comando a seguir para iniciar.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 6

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 6

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Instale o OpenSSL Dynamic Engine AWS CloudHSM para Client SDK 3

As etapas a seguir descrevem como instalar e configurar o mecanismo AWS CloudHSM dinâmico para OpenSSL com o Client SDK 3. Para informações sobre a atualização, consulte [Fazer upgrade do Client SDK 3](#).

Para instalar e configurar o mecanismo OpenSSL

1. Use os seguintes comandos para fazer download e instalar o mecanismo OpenSSL.

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

CentOS 6

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

RHEL 6

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-dyn_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-dyn_latest_amd64.deb
```

O mecanismo OpenSSL está instalado em `/opt/cloudhsm/lib/libcloudhsm_openssl.so`.

2. Use o comando a seguir para definir uma variável de ambiente denominada `n3fips_password` que contém as credenciais de um usuário de criptografia (CU).

```
$ export n3fips_password=<HSM user name>:<password>
```

Use o OpenSSL Dynamic Engine AWS CloudHSM para Client SDK 3

Para usar o mecanismo AWS CloudHSM dinâmico para OpenSSL a partir de um aplicativo integrado ao OpenSSL, certifique-se de que seu aplicativo use o mecanismo dinâmico OpenSSL chamado `cloudhsm`. A biblioteca compartilhada para o mecanismo dinâmico está localizada em `/opt/cloudhsm/lib/libcloudhsm_openssl.so`.

Para usar o mecanismo AWS CloudHSM dinâmico do OpenSSL a partir da linha de comando do OpenSSL, use a `-engine` opção para especificar o mecanismo dinâmico do OpenSSL chamado `cloudhsm`. Por exemplo:

```
$ openssl s_server -cert <server.crt> -key <server.key> -engine cloudhsm
```

Provedor de JCE para AWS CloudHSM Client SDK 3

O provedor AWS CloudHSM JCE é uma implementação de provedor criada a partir da estrutura do provedor Java Cryptographic Extension (JCE). O JCE permite que você execute operações criptográficas usando o Java Development Kit (JDK). Neste guia, o provedor AWS CloudHSM JCE às vezes é chamado de provedor JCE. Use o provedor JCE e o JDK para transferir operações criptográficas para o HSM.

Tópicos

- [Instale o provedor JCE para o AWS CloudHSM Client SDK 3](#)
- [Fundamentos do gerenciamento de chaves no provedor de JCE para o AWS CloudHSM Client SDK 3](#)
- [Mecanismos compatíveis com o Client SDK 3 for AWS CloudHSM Client SDK 3](#)
- [Atributos de chave Java compatíveis com o AWS CloudHSM Client SDK 3](#)
- [Exemplos de código para a biblioteca AWS CloudHSM de software para Java for Client SDK 3](#)
- [AWS CloudHSM KeyStore Classe Java para Client SDK 3](#)

Instale o provedor JCE para o AWS CloudHSM Client SDK 3

Antes de usar o provedor JCE, você precisa do AWS CloudHSM cliente.

O cliente é um daemon que estabelece comunicação end-to-end criptografada com o HSMs em seu cluster. O provedor JCE se comunica localmente com o cliente. Se você não instalou e configurou o pacote do AWS CloudHSM cliente, faça isso agora seguindo as etapas em [Instalar o cliente \(Linux\)](#). Após instalar e configurar o cliente, use o comando a seguir para iniciá-lo.

O provedor JCE é suportado somente em Linux e sistemas operacionais compatíveis.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

CentOS 8

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 8

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 20.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Use as seções a seguir para instalar, validar e fornecer credenciais ao provedor.

Tópicos

- [Etapa 1: instalar o provedor JCE](#)
- [Etapa 2: validar a instalação](#)
- [Etapa 3: fornecer credenciais ao provedor JCE](#)

Etapa 1: instalar o provedor JCE

Use os seguintes comandos para fazer download e instalar o provedor JCE. Este provedor só tem suporte no Linux e em sistemas operacionais compatíveis.

Note

Para atualizar, consulte [Fazer upgrade do Client SDK 3](#).

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-jce-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-jce_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-jce_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-jce_latest_u18.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-jce_latest_u18.04_amd64.deb
```

Depois de executar os comandos anteriores, você pode encontrar os seguintes arquivos do provedor JCE:

- /opt/cloudhsm/java/cloudhsm-*<version>*.jar
- /opt/cloudhsm/java/cloudhsm-test-*<version>*.jar
- /opt/cloudhsm/java/hamcrest-all-1.3.jar
- /opt/cloudhsm/java/junit.jar

- /opt/cloudhsm/java/log4j-api-2.17.1.jar
- /opt/cloudhsm/java/log4j-core-2.17.1.jar
- /opt/cloudhsm/lib/libcaviumjca.so

Etapa 2: validar a instalação

Execute operações básicas no HSM para validar a instalação.

Para validar a instalação do provedor JCE

1. (Opcional) Se você ainda não tiver o Java instalado em seu ambiente, use o comando a seguir para instalá-lo.

Linux (and compatible libraries)

```
$ sudo yum install java-1.8.0-openjdk
```

Ubuntu

```
$ sudo apt-get install openjdk-8-jre
```

2. Use os comandos a seguir para definir as variáveis de ambiente necessárias. Substitua *<HSM user name>* e *<password>* pelas credenciais de um usuário criptográfico (UC).

```
$ export LD_LIBRARY_PATH=/opt/cloudhsm/lib
```

```
$ export HSM_PARTITION=PARTITION_1
```

```
$ export HSM_USER=<HSM user name>
```

```
$ export HSM_PASSWORD=<password>
```

3. Use o comando a seguir para executar o teste de funcionalidade básica. Se bem-sucedido, a saída do comando deverá ser semelhante à saída a seguir.

```
$ java8 -classpath "/opt/cloudhsm/java/*" org.junit.runner.JUnitCore  
TestBasicFunctionality
```



```
JUnit version 4.11
.2018-08-20 17:53:48,514 DEBUG [main] TestBasicFunctionality
  (TestBasicFunctionality.java:33) - Adding provider.
2018-08-20 17:53:48,612 DEBUG [main] TestBasicFunctionality
  (TestBasicFunctionality.java:42) - Logging in.
2018-08-20 17:53:48,612 INFO [main] cfm2.LoginManager (LoginManager.java:104) -
  Looking for credentials in HsmCredentials.properties
2018-08-20 17:53:48,612 INFO [main] cfm2.LoginManager (LoginManager.java:122) -
  Looking for credentials in System.properties
2018-08-20 17:53:48,613 INFO [main] cfm2.LoginManager (LoginManager.java:130) -
  Looking for credentials in System.env
  SDK Version: 2.03
2018-08-20 17:53:48,655 DEBUG [main] TestBasicFunctionality
  (TestBasicFunctionality.java:54) - Generating AES Key with key size 256.
2018-08-20 17:53:48,698 DEBUG [main] TestBasicFunctionality
  (TestBasicFunctionality.java:63) - Encrypting with AES Key.
2018-08-20 17:53:48,705 DEBUG [main] TestBasicFunctionality
  (TestBasicFunctionality.java:84) - Deleting AES Key.
2018-08-20 17:53:48,707 DEBUG [main] TestBasicFunctionality
  (TestBasicFunctionality.java:92) - Logging out.

Time: 0.205

OK (1 test)
```

Etapa 3: fornecer credenciais ao provedor JCE

HSMs precisa autenticar seu aplicativo Java antes que o aplicativo possa usá-los. Cada aplicativo pode usar uma sessão. HSMs autentique uma sessão usando o método de login explícito ou login implícito.

Login explícito: esse método permite que você forneça credenciais do CloudHSM diretamente no aplicativo. Ele usa o método `LoginManager.login()`, em que você passa no nome do usuário, a senha e o ID da partição do HSM do usuário CU. Para obter mais informações sobre como usar o método de login explícito, consulte o código de exemplo [Fazer login em um HSM](#).

Login implícito: esse método permite que você defina as credenciais do CloudHSM em um novo arquivo de propriedades, propriedades do sistema, ou como variáveis de ambiente.

- Novo arquivo de propriedades: crie um novo arquivo chamado `HsmCredentials.properties` e adicione-o ao CLASSPATH de seu aplicativo. O arquivo deve conter o seguinte:

```
HSM_PARTITION = PARTITION_1  
HSM_USER = <HSM user name>  
HSM_PASSWORD = <password>
```

- Propriedades do sistema: defina as credenciais por meio das propriedades do sistema ao executar seu aplicativo. Os exemplos a seguir mostram duas formas diferentes de fazer isso:

```
$ java -DHSM_PARTITION=PARTITION_1 -DHSM_USER=<HSM user name> -  
DHSM_PASSWORD=<password>
```

```
System.setProperty("HSM_PARTITION", "PARTITION_1");  
System.setProperty("HSM_USER", "<HSM user name>");  
System.setProperty("HSM_PASSWORD", "<password>");
```

- Variáveis de ambiente: defina as credenciais como variáveis de ambiente.

```
$ export HSM_PARTITION=PARTITION_1  
$ export HSM_USER=<HSM user name>  
$ export HSM_PASSWORD=<password>
```

As credenciais talvez não estejam disponíveis se o aplicativo não fornecê-las ou se você tentar uma operação antes que o HSM autentique a sessão. Nesses casos, a biblioteca de software do CloudHSM para Java procura as credenciais na seguinte ordem:

1. `HsmCredentials.properties`
2. Propriedades do sistema
3. Variáveis de ambiente

Tratamento de erros

O tratamento de erros é mais fácil com o login explícito do que com o método de login implícito. Ao usar a classe `LoginManager`, você tem mais controle sobre como seu aplicativo trata as falhas. O método de login implícito dificulta a compreensão do tratamento de erros quando as credenciais são inválidas ou quando HSMs há problemas na sessão de autenticação.

Fundamentos do gerenciamento de chaves no provedor de JCE para o AWS CloudHSM Client SDK 3

Os conceitos básicos do gerenciamento de chaves no provedor JCE envolvem a importação e a exportação, o carregamento por identificador ou a exclusão de chaves. Para obter mais informações sobre como gerenciar chaves, consulte [Gerenciar chaves](#) no exemplo de código.

Você também pode encontrar mais amostras de código de provedor JCE em [Exemplos de código](#).

Mecanismos compatíveis com o Client SDK 3 for AWS CloudHSM Client SDK 3

Este tópico fornece informações sobre os mecanismos suportados pelo provedor JCE com o AWS CloudHSM Client SDK 3. Para obter informações sobre as interfaces e classes de mecanismo da Java Cryptography Architecture (JCA) suportadas pelo AWS CloudHSM, consulte os tópicos a seguir.

Tópicos

- [Chaves compatíveis](#)
- [Criptografias compatíveis](#)
- [Resumos compatíveis](#)
- [Algoritmos de código de autenticação de mensagens por hash \(HMAC\) compatíveis](#)
- [sign/verify Mecanismos suportados](#)
- [Anotações do mecanismo](#)

Chaves compatíveis

A biblioteca de AWS CloudHSM software para Java permite gerar os seguintes tipos de chaves.

- AES: chaves AES de 128, 192 e 256 bits.
- DESede — Chave 3DES de 92 bits. Consulte a nota [1](#) abaixo para ver uma mudança futura.
- Pares de chaves do ECC para curvas NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1 (Blockchain).
- RSA: 2.048 bits para chaves RSA de 4.096 bits, em incrementos de 256 bits.

Além dos parâmetros padrão, oferecemos suporte aos seguintes parâmetros para cada chave gerada.

- **Label:** Um rótulo de chave que você pode usar para procurar chaves.
- **isExtractable:** Indica se a chave pode ser exportada do HSM.
- **isPersistent:** Indica se a chave permanece no HSM ao término da sessão atual.

 Note

A versão 3.1 da biblioteca Java fornece a capacidade de especificar parâmetros em maior detalhes. Para obter mais informações, consulte [Atributos Java compatíveis](#).

Criptografias compatíveis

A biblioteca AWS CloudHSM de software para Java suporta as seguintes combinações de algoritmo, modo e preenchimento.

Algoritmo	Modo	Padding	Observações
AES	CBC	AES/CBC/N oPadding AES/CBC/P KCS5Padding	Implementa Cipher.EN CRYPT_MODE e Cipher.DE CRYPT_MODE .
AES	ECB	AES/ECB/N oPadding AES/ECB/P KCS5Padding	Implementa Cipher.EN CRYPT_MODE e Cipher.DE CRYPT_MODE . Use a transformação de AES.
AES	CTR	AES/CTR/N oPadding	Implementa Cipher.EN CRYPT_MODE e Cipher.DE CRYPT_MODE .

Algoritmo	Modo	Padding	Observações
AES	GCM	AES/GCM/NoPadding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE, Cipher.WRAP_MODE e Cipher.UNWRAP_MODE. Ao executar a criptografia AES-GCM, o HSM ignora o vetor de inicialização (IV) na solicitação e usa um IV que ele mesmo gera. Quando a operação for concluída, você deverá chamar Cipher.getIV() para obter o IV.
AESWrap	ECB	AESWrap/ECB/ZeroPadding AESWrap/ECB/NoPadding AESWrap/ECB/PKCS5Padding	Implementa Cipher.WRAP_MODE e Cipher.UNWRAP_MODE. Use a transformação de AES.

Algoritmo	Modo	Padding	Observações
DESede (DES triplo)	CBC	DESede/CBC/ NoPadding DESede/CBC/ PKCS5Padding	Implementa <code>Cipher.ENCRYPT_MODE</code> e <code>Cipher.DECRYPT_MODE</code>. As rotinas de geração de chaves aceitam um tamanho de 168 ou 192 bits. No entanto, internamente, todas as DESede chaves são de 192 bits. Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações
DESede (DES triplo)	ECB	DESede/ECB/ NoPadding DESede/ECB/ PKCS5Padding	Implementa <code>Cipher.ENCRYPT_MODE</code> e <code>Cipher.DECRYPT_MODE</code>. As rotinas de geração de chaves aceitam um tamanho de 168 ou 192 bits. No entanto, internamente, todas as DESede chaves são de 192 bits. Consulte a nota 1 abaixo para ver uma mudança futura.
RSA	ECB	RSA/ECB/ NoPadding RSA/ECB/ PKCS1Padding	Implementa <code>Cipher.ENCRYPT_MODE</code> e <code>Cipher.DECRYPT_MODE</code>. Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações
RSA	ECB	RSA/ECB/0 AEPPadding	Implementa Cipher.EN CRYPT_MOD E , Cipher.DE CRYPT_MOD E , Cipher.WR AP_MODE e Cipher.UN WRAP_MODE .
		RSA/ECB/0 AEPWithSH A-1ANDMGF 1Padding	
		RSA/ECB/0 AEPWithSH A-224ANDM GF1Padding	0AEPPadding é 0AEP com o tipo de preenchimento SHA-1.
		RSA/ECB/0 AEPWithSH A-256ANDM GF1Padding	
		RSA/ECB/0 AEPWithSH A-384ANDM GF1Padding	
		RSA/ECB/0 AEPWithSH A-512ANDM GF1Padding	
RSAAESWrap	ECB	0AEPPADDING	Implementa Cipher.WR AP_Mode e Cipher.UN WRAP_MODE .

Resumos compatíveis

A biblioteca AWS CloudHSM de software para Java suporta os seguintes resumos de mensagens.

- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512

Note

Os dados com extensão inferior a 16 KB são criptografadas no HSM, enquanto nos dados maiores se usa hash localmente no software.

Algoritmos de código de autenticação de mensagens por hash (HMAC) compatíveis

A biblioteca AWS CloudHSM de software para Java suporta os seguintes algoritmos HMAC.

- HmacSHA1
- HmacSHA224
- HmacSHA256
- HmacSHA384
- HmacSHA512

sign/verify Mecanismos suportados

A biblioteca AWS CloudHSM de software para Java suporta os seguintes tipos de assinatura e verificação.

Tipos de assinatura RSA

- NONEwithRSA
- SHA1withRSA
- SHA224withRSA

- SHA256withRSA
- SHA384withRSA
- SHA512withRSA
- SHA1withRSA/PSS
- SHA224withRSA/PSS
- SHA256withRSA/PSS
- SHA384withRSA/PSS
- SHA512withRSA/PSS

Tipos de assinatura ECDSA

- NONEwithECDSA
- SHA1withECDSA
- SHA224withECDSA
- SHA256withECDSA
- SHA384withECDSA
- SHA512withECDSA

Anotações do mecanismo

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Atributos de chave Java compatíveis com o AWS CloudHSM Client SDK 3

Este tópico descreve como usar uma extensão proprietária para a biblioteca Java versão 3.1 para definir os principais atributos AWS CloudHSM do Client SDK 3. Use essa extensão para definir atributos de chave compatíveis e seus valores durante estas operações:

- Geração de chaves
- Importação de chaves
- Desencapsulamento de chaves

Note

A extensão para definir atributos de chave personalizados é um recurso opcional. Se você já tem o código que funciona na biblioteca Java versão 3.0, não é necessário modificá-lo. As chaves criadas continuarão contendo os mesmos atributos de antes.

Tópicos

- [Noções básicas sobre atributos](#)
- [Atributos do compatíveis](#)
- [Definir atributos para uma chave](#)
- [Reunir todos os componentes](#)

Noções básicas sobre atributos

Use atributos para especificar quais ações são permitidas em objetos de chave, incluindo chaves públicas, privadas ou secretas. Defina os atributos e valores de chave durante as operações de criação de objetos de chave.

No entanto, a Java Cryptography Extension (JCE) não especifica como você deve definir valores em atributos de chave, portanto, a maioria das ações era permitida por padrão. Em contrapartida, o padrão PKCS #11 define um conjunto de atributos abrangente com padrões mais restritivos. A partir da biblioteca Java versão 3.1, o CloudHSM fornece uma extensão proprietária que permite definir valores mais restritivos para atributos usados frequentemente.

Atributos do compatíveis

É possível definir valores para atributos listados na tabela abaixo. Como melhor prática, defina valores somente para os atributos que deseja tornar restritivos. Se você não especificar um valor, o CloudHSM utilizará o valor padrão especificado na tabela abaixo. Uma célula vazia nas colunas Valor padrão indica que não há nenhum valor padrão específico atribuído ao atributo.

Atributo	Valor padrão		Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves

Atributo	Valor padrão			Observações
CKA_TOKEN	FALSE	FALSE	FALSE	Uma chave permanente que é replicada HSMs em todo o cluster e incluída nos backups. CKA_TOKEN = FALSE requer uma chave de sessão, que é carregada somente em um HSM e apagada automaticamente quando a conexão com o HSM é interrompida.
CKA_LABEL				Uma string definida pelo usuário. Ela permite identificar chaves no HSM de forma conveniente.
CKA_EXPORTABLE	TRUE		TRUE	True indica que você pode exportar essa chave para fora do HSM.

Atributo	Valor padrão			Observações
CKA_ENCRYPT	TRUE	TRUE		True indica que você pode usar a chave para criptografar qualquer buffer.
CKA_DECRYPT	TRUE		TRUE	True indica que você pode usar a chave para descriptografar qualquer buffer. Geralmente, você define isso como FALSE para uma chave cujo CKA_WRAP esteja definido como true.
CKA_WRAP	TRUE	TRUE		True indica que você pode usar a chave para encapsular outra chave. Geralmente, isso é definido como FALSE para chaves privadas.
CKA_UNWRAP	TRUE		TRUE	True indica que você pode usar a chave para desencapsular (importar) outra chave.

Atributo	Valor padrão			Observações
CKA_SIGN	TRUE		TRUE	True indica que você pode usar a chave para assinar um resumo de mensagens . Geralment e, é definido como FALSE para chaves públicas e para chaves privadas arquivadas.
CKA_VERIFY	TRUE	TRUE		True indica que você pode usar a chave para verificar uma assinatura. Isso geralmente é definido como FALSE para chaves privadas.

Atributo	Valor padrão			Observações
CKA_PRIVATE	TRUE	TRUE	TRUE	True indica que um usuário pode não acessar a chave até que o usuário seja autenticado. Para maior clareza, os usuários não podem acessar as chaves no CloudHSM até que sejam autenticados, mesmo se esse atributo estiver definido como FALSE.

 Note

Você obterá maior suporte para atributos na biblioteca PKCS #11. Para obter mais informações, consulte [Atributos PKCS #11 compatíveis](#).

Definir atributos para uma chave

O `CloudHsmKeyAttributesMap` é um objeto semelhante ao [Mapa Java](#), que pode ser usado para definir valores de atributos para objetos de chave. Os métodos para a função `CloudHsmKeyAttributesMap` funcionam de forma semelhante aos métodos usados na manipulação de mapa Java.

Para definir valores personalizados, existem duas opções:

- Usar os métodos listados na tabela a seguir

- Usar padrões do construtor demonstrados posteriormente nesse documento.

Os objetos de mapa de atributos oferecem suporte para os seguintes métodos para definir atributos:

Operation	Valor de retorno	Método CloudHSMKeyAttributesMap
Obter o valor de um atributo de chave para uma chave existente	Objeto (contendo o valor) ou nulo	<code>get(keyAttribute)</code>
Preencher o valor de um atributo de chave	O valor anterior associado a um atributo de chave, ou nulo se não houver mapeamento para um atributo de chave	<code>put(keyAttribute, valor)</code>
Preencher valores para múltiplos atributos de chave	N/D	<code>putAll () keyAttributesMap</code>
Remover um par de valor-chave do mapa de atributos	O valor anterior associado a um atributo de chave, ou nulo se não houver mapeamento para um atributo de chave	<code>remove(keyAttribute)</code>

Note

Todos os atributos que você não especificar explicitamente serão definidos como os padrões listados na tabela anterior em [the section called “Atributos do compatíveis”](#).

Exemplo de padrão do construtor

Geralmente, para os desenvolvedores será mais conveniente usar classes por meio do padrão do Construtor. Conforme os exemplos:

```
import com.amazonaws.cloudhsm.CloudHsmKeyAttributes;
import com.amazonaws.cloudhsm.CloudHsmKeyAttributesMap;
import com.amazonaws.cloudhsm.CloudHsmKeyPairAttributesMap;
```



```
CloudHsmKeyAttributesMap keyAttributesSessionDecryptionKey =
    new CloudHsmKeyAttributesMap.Builder()
        .put(CloudHsmKeyAttributes.CKA_LABEL, "ExtractableSessionKeyEncryptDecrypt")
        .put(CloudHsmKeyAttributes.CKA_WRAP, false)
        .put(CloudHsmKeyAttributes.CKA_UNWRAP, false)
        .put(CloudHsmKeyAttributes.CKA_SIGN, false)
        .put(CloudHsmKeyAttributes.CKA_VERIFY, false)
        .build();

CloudHsmKeyAttributesMap keyAttributesTokenWrappingKey =
    new CloudHsmKeyAttributesMap.Builder()
        .put(CloudHsmKeyAttributes.CKA_LABEL, "TokenWrappingKey")
        .put(CloudHsmKeyAttributes.CKA_TOKEN, true)
        .put(CloudHsmKeyAttributes.CKA_ENCRYPT, false)
        .put(CloudHsmKeyAttributes.CKA_DECRYPT, false)
        .put(CloudHsmKeyAttributes.CKA_SIGN, false)
        .put(CloudHsmKeyAttributes.CKA_VERIFY, false)
        .build();
```

Os desenvolvedores também podem utilizar conjuntos de atributos predefinidos como uma forma conveniente para aplicar as melhores práticas em modelos de chave. Exemplo:

```
//best practice template for wrapping keys

CloudHsmKeyAttributesMap commonKeyAttrs = new CloudHsmKeyAttributesMap.Builder()
    .put(CloudHsmKeyAttributes.CKA_EXTRACTABLE, false)
    .put(CloudHsmKeyAttributes.CKA_DECRYPT, false)
    .build();

// initialize a new instance of CloudHsmKeyAttributesMap by copying commonKeyAttrs
// but with an appropriate label

CloudHsmKeyAttributesMap firstKeyAttrs = new CloudHsmKeyAttributesMap(commonKeyAttrs);
firstKeyAttrs.put(CloudHsmKeyAttributes.CKA_LABEL, "key label");

// alternatively, putAll() will overwrite existing values to enforce conformance

CloudHsmKeyAttributesMap secondKeyAttrs = new CloudHsmKeyAttributesMap();
secondKeyAttrs.put(CloudHsmKeyAttributes.CKA_DECRYPT, true);
secondKeyAttrs.put(CloudHsmKeyAttributes.CKA_ENCRYPT, true);
secondKeyAttrs.put(CloudHsmKeyAttributes.CKA_LABEL, "safe wrapping key");
secondKeyAttrs.putAll(commonKeyAttrs); // will overwrite CKA_DECRYPT to be FALSE
```

Definir atributos para um par de chaves

Use a classe Java `CloudHsmKeyPairAttributesMap` para manipular atributos de chave para um par de chaves. O `CloudHsmKeyPairAttributesMap` encapsula dois objetos `CloudHsmKeyAttributesMap`; um para uma chave pública e outro para uma chave privada.

Para definir atributos individuais para a chave pública e privada separadamente, é possível usar o método `put()` no objeto de mapa `CloudHsmKeyAttributes` correspondente para essa chave. Use o método `getPublic()` para recuperar o mapa de atributos para a chave pública e use `getPrivate()` para recuperar o mapa de atributos para a chave privada. Preencha o valor de múltiplos atributos de chave para os pares de chaves públicas e privadas usando `putAll()` com um mapa de atributos de um par de chaves como argumento.

Exemplo de padrão do construtor

Geralmente, para os desenvolvedores será mais conveniente definir atributos de chave usando o padrão do Construtor. Por exemplo:

```
import com.amazonaws.cloudhsm.CloudHsmKeyAttributes;
import com.amazonaws.cloudhsm.CloudHsmKeyAttributesMap;
import com.amazonaws.cloudhsm.CloudHsmKeyPairAttributesMap;

//specify attributes up-front
CloudHsmKeyAttributesMap keyAttributes =
    new CloudHsmKeyAttributesMap.Builder()
        .put(CloudHsmKeyAttributes.CKA_SIGN, false)
        .put(CloudHsmKeyAttributes.CKA_LABEL, "PublicCertSerial12345")
        .build();

CloudHsmKeyPairAttributesMap keyPairAttributes =
    new CloudHsmKeyPairAttributesMap.Builder()
        .withPublic(keyAttributes)
        .withPrivate(
            new CloudHsmKeyAttributesMap.Builder() //or specify them inline
                .put(CloudHsmKeyAttributes.CKA_LABEL, "PrivateCertSerial12345")
                .put(CloudHsmKeyAttributes.CKA_WRAP, FALSE)
                .build()
        )
        .build();
```

Note

[Para obter mais informações sobre essa extensão proprietária, consulte o arquivo Javadoc e a amostra em.](#) [GitHub](#) Para explorar o Javadoc, faça download do arquivo e expanda-o.

Reunir todos os componentes

Para especificar os atributos de chave com as suas operações de chave, siga estes passos:

1. Instancie `CloudHsmKeyAttributesMap` para chaves simétricas ou `CloudHsmKeyPairAttributesMap` para par de chaves.
2. Defina o objeto de atributos do passo 1 com os atributos e valores de chave necessários.
3. Instancie uma classe `Cavium*ParameterSpec`, correspondente ao tipo de chave específico e passe este objeto de atributos configurados para o construtor.
4. Passe esse objeto `Cavium*ParameterSpec` para uma classe ou método de criptografia correspondente.

Para referência, a tabela a seguir contém as classes e os métodos `Cavium*ParameterSpec` que oferecem suporte aos atributos de chave personalizados.

Tipo de chave	Classe de especificação de parâmetros	Exemplo de construtores
Classe base	<code>CaviumKeyGenAlgorithmParameterSpec</code>	<code>CaviumKeyGenAlgorithmParameterSpec(CloudHsmKeyAttributesMap keyAttributesMap)</code>
DES	<code>CaviumDESKeyGenParameterSpec</code>	<code>CaviumDESKeyGenParameterSpec(int keySize, byte[] iv, CloudHsmKeyAttributesMap keyAttributesMap)</code>

Tipo de chave	Classe de especificação de parâmetros	Exemplo de construtores
RSA	<code>CaviumRSAKeyGenParameterSpec</code>	<code>CaviumRSAKeyGenParameterSpec(int keysize, BigInteger publicExponent, CloudHsmKeyPairAttributesMap keyPairAttributesMap)</code>
Secret	<code>CaviumGenericSecretKeyGenParameterSpec</code>	<code>CaviumGenericSecretKeyGenParameterSpec(int size, CloudHsmKeyAttributesMap keyAttributesMap)</code>
AES	<code>CaviumAESKeyGenParameterSpec</code>	<code>CaviumAESKeyGenParameterSpec(int keySize, byte[] iv, CloudHsmKeyAttributesMap keyAttributesMap)</code>
EC	<code>CaviumECGenParameterSpec</code>	<code>CaviumECGenParameterSpec(String stdName, CloudHsmKeyPairAttributesMap keyPairAttributesMap)</code>

Código de exemplo: gerar e encapsular uma chave

Estes exemplos de código demonstram as etapas para duas operações diferentes: geração de chaves e encapsulamento de chaves:

```
// Set up the desired key attributes

KeyGenerator keyGen = KeyGenerator.getInstance("AES", "Cavium");
CaviumAESKeyGenParameterSpec keyAttributes = new CaviumAESKeyGenParameterSpec(
    256,
    new CloudHsmKeyAttributesMap.Builder()
        .put(CloudHsmKeyAttributes.CKA_LABEL, "MyPersistentAESKey")
        .put(CloudHsmKeyAttributes.CKA_EXTRACTABLE, true)
        .put(CloudHsmKeyAttributes.CKA_TOKEN, true)
        .build()
);

// Assume we already have a handle to the myWrappingKey
// Assume we already have the wrappedBytes to unwrap

// Unwrap a key using Custom Key Attributes

CaviumUnwrapParameterSpec unwrapSpec = new
    CaviumUnwrapParameterSpec(myInitializationVector, keyAttributes);

Cipher unwrapCipher = Cipher.getInstance("AESWrap", "Cavium");
unwrapCipher.init(Cipher.UNWRAP_MODE, myWrappingKey, unwrapSpec);
Key unwrappedKey = unwrapCipher.unwrap(wrappedBytes, "AES", Cipher.SECRET_KEY);
```

Exemplos de código para a biblioteca AWS CloudHSM de software para Java for Client SDK 3

Este tópico fornece recursos e informações sobre exemplos de código Java para o Client SDK 3 do AWS CloudHSM .

Pré-requisitos

Antes de executar as amostras, você deve configurar seu ambiente:

- Instale e configure o [provedor Java Cryptographic Extension \(JCE – extensão de criptografia Java\)](#) e o [pacote do cliente do AWS CloudHSM](#).
- Configure um [nome de usuário e senha de HSM](#) válidos. As permissões do usuário de criptografia (CU) são suficientes para essas tarefas. O aplicativo usa essas credenciais para fazer login no HSM em cada exemplo.
- Decida como fornecer credenciais ao provedor [JCE](#).

Exemplos de código

Os exemplos de código a seguir mostram como usar o [provedor JCE AWS CloudHSM](#) para realizar tarefas básicas. Mais exemplos de código estão disponíveis em [GitHub](#).

- [Log in to an HSM](#)
- [Gerenciar chaves](#)
- [Gerar uma chave de AES](#).
- [Encrypt and decrypt with AES-GCM](#)
- [Encrypt and decrypt with AES-CTR](#)
- [Encrypt and decrypt with D3DES-ECB](#)^{veja nota 1}
- [Wrap and unwrap keys with AES-GCM](#)
- [Encapsular e desencapsular chaves com AES](#)
- [Encapsular e desencapsular chaves com RSA](#)
- [Use supported key attributes](#)
- [Enumerate keys in the key store](#)
- [Use the CloudHSM key store](#)
- [Assinar mensagens em um exemplo encadeado várias vezes](#)
- [Assinatura e verificação com chaves EC](#)

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

AWS CloudHSM KeyStore Classe Java para Client SDK 3

A AWS CloudHSM **KeyStore** classe fornece um armazenamento de PKCS12 chaves para fins especiais que permite o acesso às AWS CloudHSM chaves por meio de aplicativos como keytool e jarsigner. Este repositório de chaves pode armazenar certificados junto com os seus dados de chave e correlacioná-los com os dados da chave armazenados no AWS CloudHSM.

 Note

Como os certificados são informações públicas e, para maximizar a capacidade de armazenamento de chaves criptográficas, AWS CloudHSM não oferece suporte ao armazenamento de certificados em HSMs.

A AWS CloudHSM KeyStore classe implementa a KeyStore Service Provider Interface (SPI) da Java Cryptography Extension (JCE). Para obter mais informações sobre o usoKeyStore, consulte [Classe KeyStore](#).

Escolha o armazenamento de chaves apropriado para o AWS CloudHSM Client SDK 3

O provedor AWS CloudHSM Java Cryptographic Extension (JCE) vem com um armazenamento de chaves padrão de passagem e somente leitura que passa todas as transações para o HSM. Esse armazenamento de chaves padrão é diferente do de propósito especial AWS CloudHSM KeyStore. Na maioria das situações, você obterá melhor desempenho de runtime e throughput usando o padrão. Você só deve usar o AWS CloudHSM KeyStore para aplicativos em que precise de suporte para certificados e operações baseadas em certificados, além de transferir as principais operações para o HSM.

Embora ambos os armazenamentos de chaves usem o provedor JCE para operações, eles são entidades independentes e não trocam informações entre si.

Carregue o repositório de chaves padrão para seu aplicativo Java da seguinte forma:

```
KeyStore ks = KeyStore.getInstance("Cavium");
```

Carregue o CloudHSM para fins especiais da seguinte forma: KeyStore

```
KeyStore ks = KeyStore.getInstance("CloudHSM")
```

Inicialize o AWS CloudHSM KeyStore for Client SDK 3

Faça login AWS CloudHSM KeyStore da mesma forma que você faz login no provedor JCE. Você pode usar variáveis de ambiente ou o arquivo de propriedades do sistema e deve fazer login antes de começar a usar o CloudHSM KeyStore. Para obter um exemplo de login em um HSM usando o JCE, consulte [Login em um HSM](#).

Se desejar, você pode especificar uma senha para criptografar o PKCS12 arquivo local que contém os principais dados do armazenamento. Ao criar o AWS CloudHSM Keystore, você define a senha e a fornece ao usar os métodos `load`, `set` e `get`.

Instancie um novo objeto CloudHSM da seguinte forma: `KeyStore`

```
ks.load(null, null);
```

Grave dados de repositório de chaves em um arquivo usando o método `store`. A partir desse ponto, você pode carregar o repositório de chaves existente usando o método `load` com o arquivo de origem e a senha da seguinte forma:

```
ks.load(inputStream, password);
```

Use o AWS CloudHSM KeyStore for AWS CloudHSM Client SDK 3

[Um objeto KeyStore CloudHSM geralmente é usado por meio de um aplicativo de terceiros, como jarsigner ou keytool.](#) Você também pode acessar o objeto diretamente com código.

AWS CloudHSM KeyStore está em conformidade com a KeyStore especificação da [classe](#) JCE e fornece as seguintes funções.

- `load`

Carrega o repositório de chaves do fluxo de entrada fornecido. Se uma senha foi definida ao salvar o repositório de chaves, essa mesma senha deve ser fornecida para que o carregamento seja bem-sucedido. Defina ambos os parâmetros como `null` para inicializar um novo repositório de chaves vazio.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");  
ks.load(inputStream, password);
```

- `aliases`

Retorna uma enumeração dos nomes de alias de todas as entradas na instância de repositório de chaves dada. Os resultados incluem objetos armazenados localmente no PKCS12 arquivo e objetos residentes no HSM.

Código de exemplo:

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
```



```
for(Enumeration<String> entry = ks.aliases(); entry.hasMoreElements();)
{
    String label = entry.nextElement();
    System.out.println(label);
}
```

- **ContainsAlias**

Retorna true se o repositório de chaves tiver acesso a pelo menos um objeto com o alias especificado. O armazenamento de chaves verifica os objetos armazenados localmente no PKCS12 arquivo e os objetos residentes no HSM.

- **DeleteEntry**

Exclui uma entrada de certificado do PKCS12 arquivo local. A exclusão de dados-chave armazenados em um HSM não é suportada usando o. AWS CloudHSM KeyStore Você pode excluir chaves com a ferramenta [key_mgmt_util](#) do CloudHSM.

- **GetCertificate**

Retorna o certificado associado a um alias, se disponível. Se o alias não existir ou fizer referência a um objeto que não for um certificado, a função retornará NULL.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
Certificate cert = ks.getCertificate(alias)
```

- **GetCertificateAlias**

Retorna o nome (alias) da primeira entrada de repositório de chaves cujos dados correspondem ao certificado fornecido.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
String alias = ks.getCertificateAlias(cert)
```

- **GetCertificateChain**

Retorna a cadeia de certificados associada ao alias fornecido. Se o alias não existir ou fizer referência a um objeto que não for um certificado, a função retornará NULL.

- **GetCreationDate**

Retorna a data de criação da entrada identificada pelo alias fornecido. Se uma data de criação não estiver disponível, a função retornará a data em que o certificado se tornou válido.

- **GetKey**

GetKey é passado para o HSM e retorna um objeto chave correspondente ao rótulo fornecido. Ao consultar getKey diretamente o HSM, ela pode ser usada para qualquer chave no HSM, independentemente de ter sido gerada pelo. KeyStore

```
Key key = ks.getKey(keyLabel, null);
```

- **IsCertificateEntry**

Verifica se a entrada com o alias fornecido representa uma entrada de certificado.

- **IsKeyEntry**

Verifica se a entrada com o alias fornecido representa uma entrada de chave. A ação pesquisa o alias no PKCS12 arquivo e no HSM.

- **SetCertificateEntry**

Atribui o certificado fornecido ao alias fornecido. Se o alias fornecido já estiver sendo usado para identificar uma chave ou certificado, um `KeyStoreException` é lançado. Você pode usar o código JCE para obter o objeto chave e, em seguida, usar o `KeyStore SetKeyEntry` método para associar o certificado à chave.

- **SetKeyEntry com chave byte[]**

No momento, essa API não é compatível com o Client SDK 3.

- **SetKeyEntry com objeto Key**

Atribui a chave fornecida ao alias fornecido e armazena-a dentro do HSM. Se o objeto Key não for do tipo `CaviumKey`, a chave será importada para o HSM como uma chave de sessão extraível.

Se o objeto Key for do tipo `PrivateKey`, ele deve ser acompanhado por uma cadeia de certificados correspondente.

Se o alias já existir, a `SetKeyEntry` chamada lança um `KeyStoreException` e impede que a chave seja substituída. Se a chave precisar ser substituída, use KMU ou JCE para esse fim.

- **EngineSize**

Retorna o número de entradas no repositório de chaves.

- **Store**

Armazena o armazenamento de chaves no fluxo de saída fornecido como um PKCS12 arquivo e o protege com a senha fornecida. Além disso, mantém todas as chaves carregadas (que são definidas usando chamadas `setKey`).

API de criptografia: próxima geração (CNG) e provedores de armazenamento de chaves (KSP) para AWS CloudHSM

O AWS CloudHSM cliente para Windows inclui provedores de CNG e KSP.

Os provedores de armazenamento de chaves (KSPs) permitem o armazenamento e a recuperação de chaves. Por exemplo, se você adicionar a função Microsoft Active Directory Certificate Services (AD CS) ao Windows Server e optar por criar uma nova chave privada da sua autoridade de certificação (CA), poderá escolher o KSP que gerenciará o armazenamento de chaves. Quando você configurar a função AD CS, poderá escolher este KSP. Para obter mais informações, consulte [Criar CA do Windows Server](#).

Cryptography API: Next Generation (CNG) é uma API de criptografia específica para o sistema operacional Microsoft Windows. O CNG permite que os desenvolvedores usem técnicas de criptografia para proteger aplicativos baseados em Windows. Em um alto nível, a AWS CloudHSM implementação do CNG fornece as seguintes funcionalidades:

- Primitivos de criptografia: permite que você execute operações de criptografia fundamentais.
- Importação e exportação de chave: permite que você importe e exporte chaves assimétricas.
- API de proteção de dados (CNG DPAPI): permite que você facilmente criptografe e descriptografe os dados.
- Armazenamento e recuperação de chave: permite armazenar com segurança e isolar a chave privada de um par de chaves assimétricas.

Tópicos

- [Verifique os provedores de KSP e CNG para AWS CloudHSM](#)
- [Pré-requisitos para usar o Windows Client AWS CloudHSM](#)
- [Associar uma AWS CloudHSM chave a um certificado](#)
- [Exemplo de código para provedor de GNV para AWS CloudHSM](#)

Verifique os provedores de KSP e CNG para AWS CloudHSM

Os provedores KSP e CNG são instalados quando você instala o cliente Windows AWS CloudHSM. Você pode instalar o cliente seguindo as etapas em [Instalar o cliente \(Windows\)](#).

Use as seções a seguir para verificar a instalação dos provedores.

Configurar e executar o AWS CloudHSM cliente Windows

Para iniciar o cliente do Windows CloudHSM, primeiro você deve estar de acordo com o [Pré-requisitos](#). Depois, atualize os arquivos de configuração que os provedores usam e inicie o cliente concluindo as etapas abaixo. Você precisa executar essas etapas na primeira vez em que usar os provedores KSP e CNG e depois de adicionar ou remover HSMs no seu cluster. Dessa forma, AWS CloudHSM é capaz de sincronizar dados e manter a consistência em tudo HSMs no cluster.

Etapa 1: interromper o AWS CloudHSM cliente

Antes de atualizar os arquivos de configuração que os provedores usam, interrompa o AWS CloudHSM cliente. Se o cliente já estiver parado, executar o comando stop não terá efeitos.

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe stop AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

Use Ctrl + C na janela de comando em que você iniciou o AWS CloudHSM cliente.

Etapa 2: atualizar os arquivos AWS CloudHSM de configuração

Essa etapa usa o -a parâmetro da [ferramenta Configure](#) para adicionar o endereço IP da elastic network interface (ENI) de um dos HSMs no cluster ao arquivo de configuração.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM ENI IP>
```

Para obter o endereço IP ENI de um HSM em seu cluster, navegue até o AWS CloudHSM console, escolha clusters e selecione o cluster desejado. Você também pode usar a [DescribeClusters](#) operação, o comando [describe-clusters](#) ou o cmdlet. [Get-HSM2Cluster](#) PowerShell. Digite somente um endereço IP ENI. Não importa qual endereço IP ENI você utilize.

Etapa 3: iniciar o AWS CloudHSM cliente

Em seguida, inicie ou reinicie o AWS CloudHSM cliente. Quando o AWS CloudHSM cliente inicia, ele usa o endereço IP ENI em seu arquivo de configuração para consultar o cluster. Em seguida, ele adiciona os endereços IP ENI de todos HSMs no cluster ao arquivo de informações do cluster.

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Verificar os provedores KSP e CNG

Você pode usar um dos seguintes comandos para determinar quais provedores são instalados no seu sistema. Os comandos listam os provedores de KSP e CNG registrados. O cliente do AWS CloudHSM não precisa estar em execução.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\ksp_config.exe" -enum
```

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cng_config.exe" -enum
```

Para verificar se os provedores KSP e CNG estão instalados na sua instância EC2 do Windows Server, você deve ver as seguintes entradas na lista:

```
Cavium CNG Provider  
Cavium Key Storage Provider
```

Se o provedor CNG estiver ausente, execute o comando a seguir.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cng_config.exe" -register
```

Se o provedor KSP estiver ausente, execute o comando a seguir.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\ksp_config.exe" -register
```

Pré-requisitos para usar o Windows Client AWS CloudHSM

Antes de iniciar o AWS CloudHSM cliente Windows e usar os provedores KSP e CNG, você deve definir as credenciais de login para o HSM em seu sistema. Você pode definir as credenciais por meio do Gerenciador de credenciais do Windows ou da variável de ambiente do sistema. Recomendamos que você use o Gerenciador de credenciais do Windows para armazenar as credenciais. Essa opção está disponível com a versão 2.0.4 e posterior do AWS CloudHSM cliente. O uso da variável de ambiente é mais fácil de configurar, mas é menos seguro do que o uso do Gerenciador de credenciais do Windows.

Gerenciador de credenciais do Windows

Você pode usar o utilitário `set_cloudhsm_credentials` ou a interface do Gerenciador de credenciais do Windows.

- Usar o utilitário **`set_cloudhsm_credentials`**:

O utilitário `set_cloudhsm_credentials` está incluído no instalador do Windows. É possível usar esse utilitário para passar convenientemente as credenciais de login do HSM para o Gerenciador de credenciais do Windows. Se desejar compilar esse utilitário a partir da origem, use o código Python que está incluído no instalador.

1. Acesse a pasta `C:\Program Files\Amazon\CloudHSM\tools\`.
2. Execute o arquivo `set_cloudhsm_credentials.exe` com os parâmetros de nome de usuário e senha do CU.

```
set_cloudhsm_credentials.exe --username <CU USER> --password <CU PASSWORD>
```

- Usar a interface do Gerenciador de credenciais:

Você pode usar a interface do Gerenciador de credenciais para gerenciar manualmente suas credenciais.

1. Para abrir o Gerenciador de credenciais, digite `credential manager` na caixa de pesquisa na barra de tarefas e selecione Credential Manager (Gerenciador de credenciais).
2. Selecione Windows Credentials (Credenciais do Windows) para gerenciar as credenciais do Windows.
3. Selecione Add a generic credential (Adicionar uma credencial genérica) e preencha os detalhes da seguinte forma:

- Em Internet ou Network Address (Endereço da Internet ou da rede), insira o nome de destino como `cloudhsm_client`.
- Em Username (Nome de usuário) e Password (Senha), insira as credenciais do CU.
- Clique em OK.

Variáveis de ambiente do sistema

Você pode definir as variáveis de ambiente do sistema que identificam um HSM e um [crypto user \(usuário de criptografia\)](#) (CU) para seu aplicativo Windows. Você pode usar o comando [setx](#) para definir as variáveis de ambiente do sistema ou definir as variáveis de ambiente do sistema permanentes [de forma programática](#) ou na guia Advanced (Avançado) do Painel de Controle nas System Properties (Propriedades do Sistema) do Windows.

Warning

Quando você define as credenciais por meio de variáveis de ambiente do sistema, a senha fica disponível em texto simples no sistema de um usuário. Para resolver esse problema, use o Gerenciador de credenciais do Windows.

Defina as seguintes variáveis de ambiente do sistema:

`n3fips_password=<CU USERNAME>:<CU PASSWORD>`

Identifica um [crypto user \(usuário de criptografia\)](#) (CU) no HSM e fornece todas as informações de login necessárias. O aplicativo é autenticado e executado como esse CU. O aplicativo tem as permissões desse CU e pode visualizar e gerenciar apenas as chaves que o CU possui e compartilha. Para criar um novo CU, use [createUser](#). Para encontrar os existentes CUs, use [ListUsers](#).

Por exemplo:

```
setx /m n3fips_password test_user:password123
```

Associar uma AWS CloudHSM chave a um certificado

Antes de usar AWS CloudHSM chaves com ferramentas de terceiros, como as da Microsoft [SignTool](#), você deve importar os metadados da chave para o repositório de certificados local e

associar os metadados a um certificado. Para importar os metadados da chave, use o utilitário `import_key.exe` que está incluído no CloudHSM versão 3.0 e posterior. As etapas a seguir fornecem informações adicionais e saída de exemplo.

Etapa 1: Importar certificado

No Windows, você deve conseguir clicar duas vezes no certificado para importá-lo para o armazenamento de certificados local.

No entanto, se clicar duas vezes não funcionar, use a [ferramenta Microsoft Certreq](#) para importar o certificado para o gerenciador de certificados. Por exemplo:

```
certreq -accept <certificatename>
```

Se essa ação falhar e você receber o erro `Key not found`, continue para a etapa 2. Se o certificado aparecer no repositório de chaves, você concluiu a tarefa e nenhuma ação adicional será necessária.

Etapa 2: Coletar informações de identificação de certificado

Se a etapa anterior não tiver sido bem-sucedida, você precisará associar sua chave privada a um certificado. No entanto, antes de criar a associação, você deve primeiro localizar o nome exclusivo do contêiner e o número de série do certificado. Use um utilitário, como `certutil`, para exibir as informações necessárias do certificado. A saída de exemplo a seguir de `certutil` mostra o nome do contêiner e o número de série.

```
===== Certificate 1 ===== Serial Number:
72000000047f7f7a9d41851b4e000000000004Issuer: CN=Enterprise-CANotBefore: 10/8/2019
11:50
AM NotAfter: 11/8/2020 12:00 PMSubject: CN=www.example.com, OU=Certificate
Management,
O=Information Technology, L=Seattle, S=Washington, C=USNon-root CertificateCert
Hash(sha1): 7f d8 5c 00 27 bf 37 74 3d 71 5b 54 4e c0 94 20 45 75 bc 65No key
provider
information Simple container name: CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c
Unique
container name: CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c
```


Etapa 3: associar a chave AWS CloudHSM privada ao certificado

Para associar a chave ao certificado, primeiro certifique-se de [iniciar o daemon do AWS CloudHSM cliente](#). Em seguida, use `import_key.exe` (que está incluído no CloudHSM versão 3.0 e superior) para associar a chave privada ao certificado. Ao especificar o certificado, use seu nome de contêiner simples. O exemplo a seguir mostra o comando e a resposta. Esta ação copia apenas os metadados da chave, a chave permanece no HSM.

```
$> import_key.exe -RSA CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c
```

```
Successfully opened Microsoft Software Key Storage Provider : 0NCryptOpenKey failed :  
80090016
```

Etapa 4: Atualizar o armazenamento de certificados

Certifique-se de que o daemon AWS CloudHSM do cliente ainda esteja em execução. Em seguida, use o verbo de `certutil`, `-repairstore`, para atualizar o número de série do certificado. O exemplo a seguir mostra o comando e a saída. Consulte a documentação da Microsoft para obter informações sobre o [verbo de -repairstore](#).

```
C:\Program Files\Amazon\CloudHSM>certutil -f -csp "Cavium Key Storage Provider"-  
repairstore my "72000000047f7f7a9d41851b4e000000000004"  
my "Personal"  
===== Certificate 1 =====  
Serial Number: 72000000047f7f7a9d41851b4e000000000004  
Issuer: CN=Enterprise-CA  
NotBefore: 10/8/2019 11:50 AM  
NotAfter: 11/8/2020 12:00 PM  
Subject: CN=www.example.com, OU=Certificate Management, O=Information Technology,  
L=Seattle, S=Washington, C=US  
Non-root CertificateCert Hash(sha1): 7f d8 5c 00 27 bf 37 74 3d 71 5b 54 4e c0 94 20 45  
75 bc 65  
SDK Version: 3.0  
Key Container = CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c  
Provider = "Cavium Key Storage Provider"  
Private key is NOT exportableEncryption test passedCertUtil: -repairstore command  
completed successfully.
```

Depois de atualizar o número de série do certificado, você pode usar esse certificado e a chave AWS CloudHSM privada correspondente com qualquer ferramenta de assinatura de terceiros no Windows.

Exemplo de código para provedor de GNV para AWS CloudHSM

 **** Somente código de exemplo – não se destina ao uso em produção ****

Esse exemplo de código é apenas para fins de ilustração. Não execute esse código na produção.

O exemplo de código seguinte mostra como enumerar os provedores criptográficos registrados em seu sistema para encontrar o provedor de CNG instalado com o cliente CloudHSM para Windows. O exemplo também mostra como criar um par de chaves assimétricas e como usar o par de chaves para assinar os dados.

Important

Antes de executar este exemplo, você deve configurar as credenciais do HSM conforme explicado nos pré-requisitos. Para obter mais detalhes, consulte [Pré-requisitos para usar o Windows Client AWS CloudHSM](#).

```
// CloudHsmCngExampleConsole.cpp : Console application that demonstrates CNG
// capabilities.
// This example contains the following functions.
//
// VerifyProvider()           - Enumerate the registered providers and retrieve Cavium
// KSP and CNG providers.
// GenerateKeyPair()         - Create an RSA key pair.
// SignData()                - Sign and verify data.
//
#include "stdafx.h"
#include <Windows.h>

#ifndef NT_SUCCESS
#define NT_SUCCESS(Status) ((NTSTATUS)(Status) >= 0)
#endif

#define CAVIUM_CNG_PROVIDER L"Cavium CNG Provider"
#define CAVIUM_KEYSTORE_PROVIDER L"Cavium Key Storage Provider"
```

```

// Enumerate the registered providers and determine whether the Cavium CNG provider
// and the Cavium KSP provider exist.
//
bool VerifyProvider()
{
    NTSTATUS status;
    ULONG cbBuffer = 0;
    PCRYPT_PROVIDERS pBuffer = NULL;
    bool foundCng = false;
    bool foundKeystore = false;

    // Retrieve information about the registered providers.
    //  cbBuffer - the size, in bytes, of the buffer pointed to by pBuffer.
    //  pBuffer - pointer to a buffer that contains a CRYPT_PROVIDERS structure.
    status = BCryptEnumRegisteredProviders(&cbBuffer, &pBuffer);

    // If registered providers exist, enumerate them and determine whether the
    // Cavium CNG provider and Cavium KSP provider have been registered.
    if (NT_SUCCESS(status))
    {
        if (pBuffer != NULL)
        {
            for (ULONG i = 0; i < pBuffer->cProviders; i++)
            {
                // Determine whether the Cavium CNG provider exists.
                if (wcscmp(CAVIUM_CNG_PROVIDER, pBuffer->rgpszProviders[i]) == 0)
                {
                    printf("Found %S\n", CAVIUM_CNG_PROVIDER);
                    foundCng = true;
                }

                // Determine whether the Cavium KSP provider exists.
                else if (wcscmp(CAVIUM_KEYSTORE_PROVIDER, pBuffer->rgpszProviders[i]) == 0)
                {
                    printf("Found %S\n", CAVIUM_KEYSTORE_PROVIDER);
                    foundKeystore = true;
                }
            }
        }
    }
    else
    {
        printf("BCryptEnumRegisteredProviders failed with error code 0x%08x\n", status);
    }
}

```

```
}

// Free memory allocated for the CRYPT_PROVIDERS structure.
if (NULL != pBuffer)
{
    BCryptFreeBuffer(pBuffer);
}

return foundCng == foundKeystore == true;
}

// Generate an asymmetric key pair. As used here, this example generates an RSA key
pair
// and returns a handle. The handle is used in subsequent operations that use the key
pair.
// The key material is not available.
//
// The key pair is used in the SignData function.
//
NTSTATUS GenerateKeyPair(BCRYPT_ALG_HANDLE hAlgorithm, BCRYPT_KEY_HANDLE *hKey)
{
    NTSTATUS status;

    // Generate the key pair.
    status = BCryptGenerateKeyPair(hAlgorithm, hKey, 2048, 0);
    if (!NT_SUCCESS(status))
    {
        printf("BCryptGenerateKeyPair failed with code 0x%08x\n", status);
        return status;
    }

    // Finalize the key pair. The public/private key pair cannot be used until this
    // function is called.
    status = BCryptFinalizeKeyPair(*hKey, 0);
    if (!NT_SUCCESS(status))
    {
        printf("BCryptFinalizeKeyPair failed with code 0x%08x\n", status);
        return status;
    }

    return status;
}

// Sign and verify data using the RSA key pair. The data in this function is hardcoded
```

```
// and is for example purposes only.
//
NTSTATUS SignData(BCRYPT_KEY_HANDLE hKey)
{
    NTSTATUS status;
    PBYTE sig;
    ULONG sigLen;
    ULONG resLen;
    BCRYPT_PKCS1_PADDING_INFO pInfo;

    // Hardcode the data to be signed (for demonstration purposes only).
    PBYTE message = (PBYTE)"d83e7716bed8a20343d8dc6845e57447";
    ULONG messageLen = strlen((char*)message);

    // Retrieve the size of the buffer needed for the signature.
    status = BCryptSignHash(hKey, NULL, message, messageLen, NULL, 0, &sigLen, 0);
    if (!NT_SUCCESS(status))
    {
        printf("BCryptSignHash failed with code 0x%08x\n", status);
        return status;
    }

    // Allocate a buffer for the signature.
    sig = (PBYTE)HeapAlloc(GetProcessHeap(), 0, sigLen);
    if (sig == NULL)
    {
        return -1;
    }

    // Use the SHA256 algorithm to create padding information.
    pInfo.pszAlgId = BCRYPT_SHA256_ALGORITHM;

    // Create a signature.
    status = BCryptSignHash(hKey, &pInfo, message, messageLen, sig, sigLen, &resLen,
        BCRYPT_PAD_PKCS1);
    if (!NT_SUCCESS(status))
    {
        printf("BCryptSignHash failed with code 0x%08x\n", status);
        return status;
    }

    // Verify the signature.
    status = BCryptVerifySignature(hKey, &pInfo, message, messageLen, sig, sigLen,
        BCRYPT_PAD_PKCS1);
}
```

```
if (!NT_SUCCESS(status))
{
    printf("BCryptVerifySignature failed with code 0x%08x\n", status);
    return status;
}

// Free the memory allocated for the signature.
if (sig != NULL)
{
    HeapFree(GetProcessHeap(), 0, sig);
    sig = NULL;
}

return 0;
}

// Main function.
//
int main()
{
    NTSTATUS status;
    BCRYPT_ALG_HANDLE hRsaAlg;
    BCRYPT_KEY_HANDLE hKey = NULL;

    // Enumerate the registered providers.
    printf("Searching for Cavium providers...\n");
    if (VerifyProvider() == false) {
        printf("Could not find the CNG and Keystore providers\n");
        return 1;
    }

    // Get the RSA algorithm provider from the Cavium CNG provider.
    printf("Opening RSA algorithm\n");
    status = BCryptOpenAlgorithmProvider(&hRsaAlg, BCRYPT_RSA_ALGORITHM,
    CAVIUM_CNG_PROVIDER, 0);
    if (!NT_SUCCESS(status))
    {
        printf("BCryptOpenAlgorithmProvider RSA failed with code 0x%08x\n", status);
        return status;
    }

    // Generate an asymmetric key pair using the RSA algorithm.
    printf("Generating RSA Keypair\n");
    GenerateKeyPair(hRsaAlg, &hKey);
}
```

```
if (hKey == NULL)
{
    printf("Invalid key handle returned\n");
    return 0;
}
printf("Done!\n");

// Sign and verify [hardcoded] data using the RSA key pair.
printf("Sign/Verify data with key\n");
SignData(hKey);
printf("Done!\n");

// Remove the key handle from memory.
status = BCryptDestroyKey(hKey);
if (!NT_SUCCESS(status))
{
    printf("BCryptDestroyKey failed with code 0x%08x\n", status);
    return status;
}

// Close the RSA algorithm provider.
status = BCryptCloseAlgorithmProvider(hRsaAlg, NULL);
if (!NT_SUCCESS(status))
{
    printf("BCryptCloseAlgorithmProvider RSA failed with code 0x%08x\n", status);
    return status;
}

return 0;
}
```

Integrando aplicativos de terceiros com AWS CloudHSM

Alguns dos [casos de uso](#) AWS CloudHSM envolvem a integração de aplicativos de software de terceiros com o HSM em seu AWS CloudHSM cluster. Ao integrar software de terceiros com AWS CloudHSM, você pode atingir uma variedade de metas relacionadas à segurança. Os seguintes tópicos descrevem como realizar algumas dessas metas.

Tópicos

- [Melhore a segurança do seu servidor web com o SSL/TLS offload in AWS CloudHSM](#)
- [Configure o Windows Server como uma autoridade de certificação \(CA\) com AWS CloudHSM](#)
- [Criptografia de dados transparente \(TDE\) do banco de dados Oracle com AWS CloudHSM](#)
- [Use o Microsoft SignTool com AWS CloudHSM para assinar arquivos](#)
- [Integração do Java Keytool e do Jarsigner com AWS CloudHSM](#)
- [Use a Ferramenta de Geração e Edição de Manifestos da Microsoft \(Mage.exe\) com AWS CloudHSM para assinar arquivos](#)
- [Outras integrações de fornecedores terceirizados com AWS CloudHSM](#)

Melhore a segurança do seu servidor web com o SSL/TLS offload in AWS CloudHSM

Os servidores web e seus clientes (navegadores da web) podem usar os protocolos Secure Sockets Layer (SSL) ou Transport Layer Security (TLS) para confirmar a identidade do servidor Web e estabelecer uma conexão segura que envia e recebe páginas Web ou outros dados pela Internet. É normalmente conhecido como HTTPS. O servidor web usa um par de chaves pública-privada e um certificado de chave SSL/TLS pública para estabelecer uma sessão HTTPS com cada cliente. Esse processo envolve muita computação para servidores web, mas você pode transferir parte disso para o seu AWS CloudHSM cluster, o que é conhecido como aceleração SSL. O descarregamento reduz a carga computacional em seus servidores da Web e fornece segurança extra ao armazenar as chaves privadas dos servidores. HSMs

Os tópicos a seguir fornecem uma visão geral de como o SSL/TLS offload with AWS CloudHSM funciona e tutoriais para configurar o SSL/TLS offload with nas seguintes plataformas. AWS CloudHSM

[Para Linux, use o mecanismo dinâmico do OpenSSL no software de servidor web NGINX ou Apache HTTP Server](#)

Para Windows, use o software de servidor web [Serviços de Informações da Internet \(IIS\) para Windows Server](#)

Tópicos

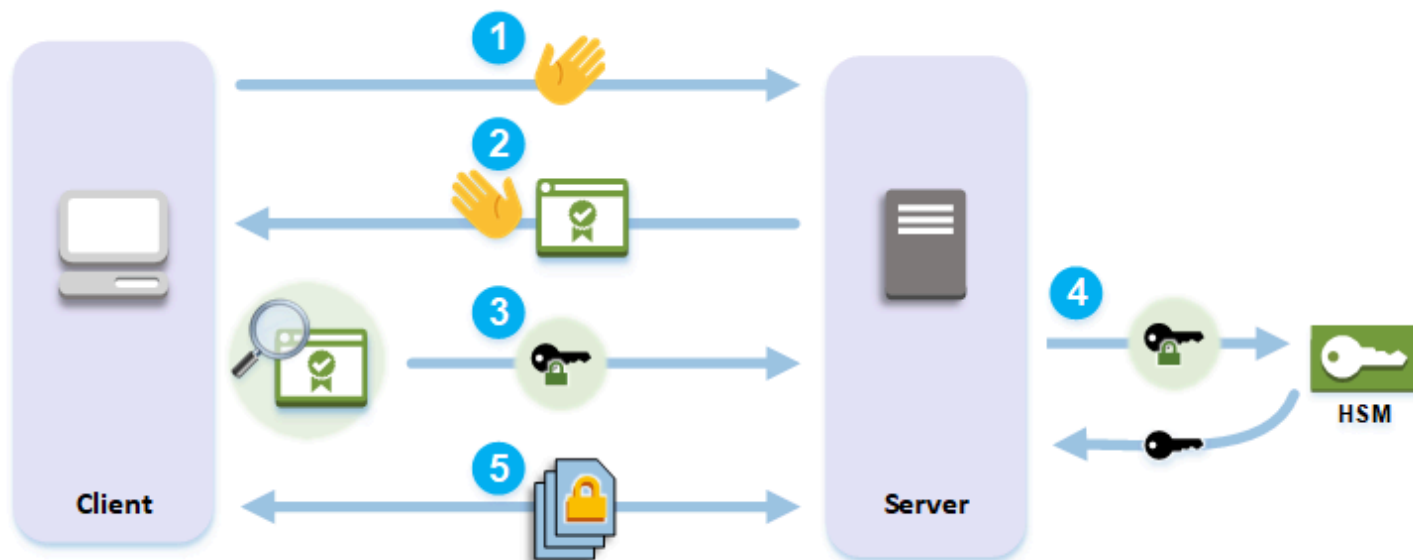
- [Como funciona a SSL/TLS descarga AWS CloudHSM](#)
- [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando NGINX ou Apache com OpenSSL](#)
- [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando NGINX ou com o OpenSSL Provider HAProxy](#)
- [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando Tomcat com JSSE](#)
- [AWS CloudHSM Descarregamento de SSL/TLS no Windows usando IIS com KSP](#)
- [Adicione um balanceador de carga com o Elastic Load Balancing AWS CloudHSM para \(opcional\)](#)

Como funciona a SSL/TLS descarga AWS CloudHSM

Para estabelecer uma conexão HTTPS, seu servidor Web executa um processo de handshake com os clientes. Como parte desse processo, o servidor transfere parte do processamento criptográfico para o AWS CloudHSM cluster, conforme mostrado HSMs na figura a seguir. Cada etapa do processo é explicada abaixo da figura.

Note

A seguinte imagem e processo supõe que RSA seja usado para verificação de servidor e troca de chaves. O processo é ligeiramente diferente quando o DiffieHellman é usado em vez do RSA.



1. O cliente envia uma mensagem de olá para o servidor.
2. O servidor responde com uma mensagem de olá e envia o certificado do servidor.
3. O cliente realiza as seguintes ações:
 - a. Verifica se o certificado SSL/TLS do servidor está assinado por um certificado raiz no qual o cliente confia.
 - b. Extrai a chave pública do certificado do servidor.
 - c. Gera um segredo pré-master e o criptografa com a chave pública do servidor.
 - d. Envia o segredo pré-master codificado para o servidor.
4. Para descriptografar o segredo pré-master do cliente, o servidor o envia ao HSM. O HSM usa a chave privada no HSM para descriptografar o segredo pré-master e, em seguida, envia o segredo pré-master ao servidor. Independentemente, o cliente e o servidor usam o segredo pré-master e algumas informações das mensagens Hello para calcular um segredo mestre.
5. O processo de handshake termina. Para o resto da sessão, todas as mensagens enviadas entre o cliente e o servidor são criptografadas com derivadas do segredo mestre.

Para saber como configurar o SSL/TLS offload com AWS CloudHSM, consulte um dos tópicos a seguir:

- [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando NGINX ou Apache com OpenSSL](#)
- [AWS CloudHSM Descarregamento de SSL/TLS no Linux usando Tomcat com JSSE](#)

- [AWS CloudHSM Descarregamento de SSL/TLS no Windows usando IIS com KSP](#)

AWS CloudHSM Descarregamento de SSL/TLS no Linux usando NGINX ou Apache com OpenSSL

Este tópico fornece step-by-step instruções para configurar o SSL/TLS offload AWS CloudHSM em um servidor web Linux.

Tópicos

- [Visão geral do](#)
- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: gerar a chave privada e o SSL/TLS certificado](#)
- [Etapa 3: configure o servidor Web](#)
- [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#)

Visão geral do

No Linux, o software de servidor Web [NGINX](#) e o [Apache HTTP Server](#) são integrados ao [OpenSSL](#) para comportar HTTPS. O [mecanismo AWS CloudHSM dinâmico do OpenSSL](#) fornece uma interface que permite que o software do servidor web use HSMs o em seu cluster para descarga criptográfica e armazenamento de chaves. O mecanismo OpenSSL é a ponte que conecta o servidor Web ao seu cluster do AWS CloudHSM .

Para concluir este tutorial, você deve primeiro escolher se deseja usar o software de servidor web NGINX ou Apache no Linux. Em seguida, este tutorial mostra como fazer o seguinte:

- Instale o software do servidor Web em uma instância do Amazon EC2.
- Configure o software do servidor Web para oferecer suporte a HTTPS com uma chave privada armazenada em seu AWS CloudHSM cluster.
- (Opcional) Use o Amazon EC2 para criar uma segunda instância de servidor Web e o Elastic Load Balancing para criar um balanceador de carga. Usar um load balanceador de carga pode aumentar o desempenho, distribuindo a carga em vários servidores. Ele também pode fornecer redundância e maior disponibilidade se um ou mais servidores falhar.

Quando estiver pronto para começar, vá para [Etapa 1: configurar os pré-requisitos](#).

Etapa 1: configurar os pré-requisitos

Plataformas diferentes exigem pré-requisitos diferentes. Use a seção de pré-requisitos abaixo que corresponde à sua plataforma.

Pré-requisitos para o Client SDK 5

Para configurar o SSL/TLS descarregamento do servidor web com o Client SDK 5, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos dois módulos de segurança de hardware (HSM)

Note

Você pode usar um único cluster HSM, mas primeiro deve desativar a durabilidade da chave do cliente. Para obter mais informações, consulte [Gerenciar configurações de durabilidade da chave do cliente](#) e [Ferramenta de configuração do Client SDK 5](#).

- Uma instância do Amazon EC2 executando um sistema operacional Linux com o seguinte software instalado:
 - Um servidor Web (NGINX ou Apache)
 - OpenSSL Dynamic Engine para Client SDK 5
- Um [usuário de criptografia](#) (CU) para ter e gerenciar a chave privada do servidor Web no HSM.

Para configurar uma instância do servidor web do Linux e criar um CU no HSM

1. Instale e configure o OpenSSL Dynamic Engine para. AWS CloudHSM Para obter mais informações sobre a instalação do OpenSSL Dynamic Engine, consulte [OpenSSL Dynamic Engine para Client SDK 5](#).
2. Em uma instância EC2 Linux que tenha acesso ao seu cluster, instale o servidor web NGINX ou Apache:

Amazon Linux

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd24 mod24_ssl
```

Amazon Linux 2

- Para obter informações sobre como baixar a versão mais recente do NGINX no Amazon Linux 2, consulte o site do [NGINX](#).

A versão mais recente do NGINX disponível para o Amazon Linux 2 usa uma versão do OpenSSL que é mais recente do que a versão do sistema do OpenSSL. Depois de instalar o NGINX, você precisa criar um link simbólico da biblioteca AWS CloudHSM OpenSSL Dynamic Engine para o local que essa versão do OpenSSL espera

```
$ sudo ln -sf /opt/cloudhsm/lib/libcloudhsm_openssl_engine.so /usr/lib64/engines-1.1/cloudhsm.so
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Amazon Linux 2023

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

CentOS 7

- Para obter informações sobre como baixar a versão mais recente do NGINX no CentOS 7, consulte o site do [NGINX](#).

A versão mais recente do NGINX disponível para o CentOS 7 usa uma versão do OpenSSL que é mais recente do que a versão do sistema do OpenSSL. Depois de instalar

o NGINX, você precisa criar um link simbólico da biblioteca AWS CloudHSM OpenSSL Dynamic Engine para o local que essa versão do OpenSSL espera

```
$ sudo ln -sf /opt/cloudhsm/lib/libcloudhsm_openssl_engine.so /usr/lib64/engines-1.1/cloudhsm.so
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Red Hat 7

- Para obter informações sobre como baixar a versão mais recente do NGINX no Red Hat 7, consulte o site do [NGINX](#).

A versão mais recente do NGINX disponível para o Red Hat 7 usa uma versão do OpenSSL que é mais recente do que a versão do sistema do OpenSSL. Depois de instalar o NGINX, você precisa criar um link simbólico da biblioteca AWS CloudHSM OpenSSL Dynamic Engine para o local que essa versão do OpenSSL espera

```
$ sudo ln -sf /opt/cloudhsm/lib/libcloudhsm_openssl_engine.so /usr/lib64/engines-1.1/cloudhsm.so
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

CentOS 8

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Red Hat 8

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Ubuntu 18.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 20.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 22.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 24.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

3. Use a CloudHSM CLI para criar um [usuário de criptografia](#). Para obter mais informações sobre o gerenciamento de usuários do HSM, consulte [Gerenciar usuários do HSM com a CLI do CloudHSM](#).

 Tip

Lembre o nome do usuário e a senha do CU. Eles serão necessários mais tarde ao gerar ou importar a chave privada HTTPS e o certificado para o servidor Web.

Depois de concluir essas etapas, vá para [Etapa 2: gerar a chave privada e o SSL/TLS certificado](#).

Observações

- Para usar o Security-Enhanced Linux (SELinux) e servidores web, você deve permitir conexões TCP de saída na porta 2223, que é a porta que o Client SDK 5 usa para se comunicar com o HSM.
- Para criar e ativar um cluster e dar acesso a uma instância do EC2 ao cluster, conclua as etapas em [Conceitos básicos de AWS CloudHSM](#). Os conceitos básicos oferecem step-by-step instruções para criar um cluster ativo com um HSM e uma instância cliente do Amazon EC2. Você pode usar essa instância de cliente como seu servidor Web.
- Para evitar a desativação da durabilidade da chave do cliente, adicione mais de um HSM ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
- Você pode usar um cliente SSH ou PuTTY para se conectar à instância do cliente. Para obter mais informações, consulte [Conectar à instância Linux utilizando SSH](#) ou [Conectar à instância Linux no Windows utilizando PuTTY](#) na documentação do Amazon C2.

Etapa 2: gerar a chave privada e o SSL/TLS certificado

Para habilitar o HTTPS, seu aplicativo de servidor web (NGINX ou Apache) precisa de uma chave privada e um certificado correspondente. Para usar o servidor web SSL/TLS offload with AWS CloudHSM, você deve armazenar a chave privada em um HSM no seu cluster. AWS CloudHSM Crie uma chave privada e use a chave para criar uma solicitação de assinatura de certificado (CSR). Você exporta uma chave privada PEM falsa do HSM, que é um arquivo de chave privada no formato PEM que contém uma referência à chave privada armazenada no HSM (não é a chave privada real). Seu servidor web usa o arquivo de chave privada PEM falso para identificar a chave privada no HSM durante SSL/TLS o descarregamento.

Gerar uma chave privada e um certificado

Gerar uma chave privada

Esta seção mostra como gerar um par de chaves usando o [CloudHSM CLI](#). Depois de gerar um par de chaves dentro do HSM, você pode exportá-lo como um arquivo PEM falso e gerar o certificado correspondente.

Instalar e configurar a CloudHSM CLI

1. [Instale e configure](#) a CloudHSM CLI.
2. Use o comando a seguir para iniciar a CloudHSM CLI.

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

3. Execute o seguinte comando para fazer login no HSM. *<user name>* Substitua pelo nome de usuário do seu usuário criptográfico

```
Command: login --username <user name> --role crypto-user
```

Gerar uma chave privada

Dependendo do seu caso de uso, você pode gerar um RSA ou um par de chaves EC. Execute um destes procedimentos:

- Para gerar uma chave privada RSA em um HSM

Use o comando [key generate-asymmetric-pair rsa](#) para gerar um par de chaves RSA. Este exemplo gera um par de chaves RSA com um módulo de 2048, um expoente

público de 65537, rótulo de chave pública de e rótulo de chave privada de *tls_rsa_pub*.
tls_rsa_private

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label tls_rsa_pub \
--private-label tls_rsa_private \
--private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "tls_rsa_pub",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
```

```

    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
      "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
      73a80fdb457aa7b20cd61e486c326e2cfd5e124a7f6a996437437812b542e3caf85928aa866f0298580f7967ee6aa
      f6e6296d6c116d5744c6d60d14d3bf3cb978fe6b75ac67b7089bafd50d8687213b31abc7dc1bad422780d29c851d5
      133022653225bd129f8491101725e9ea33e1ded83fb57af35f847e532eb30cd7e726f23910d2671c6364092e83469
      ac3160f0ca9725d38318b7",
    "modulus-size-bits": 2048
  }
},
"private_key": {
  "key-reference": "0x0000000000280cc7",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "cluster-coverage": "full"
  },
  "attributes": {
    "key-type": "rsa",
    "label": "tls_rsa_private",
    "id": "",
    "check-value": "0x01fe6e",
    "class": "private-key",
    "encrypt": false,
    "decrypt": true,
    "token": true,
    "always-sensitive": true,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,

```

```

    "private": true,
    "sensitive": true,
    "sign": true,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 1217,
    "public-exponent": "0x010001",
    "modulus":
"0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
    "modulus-size-bits": 2048
  }
}
}
}

```

- Para gerar uma chave privada EC em um HSM

Use o comando [key generate-asymmetric-pair ec](#) para gerar um par de chaves EC. Este exemplo gera um par de chaves EC com a prime256v1 curva (correspondente à NID_X9_62_prime256v1 curva), um rótulo de chave pública de *tls_ec_pub* e um rótulo de chave privada de *tls_ec_private*.

```

aws-cloudhsm > key generate-asymmetric-pair ec \
  --curve prime256v1 \
  --public-label tls_ec_pub \
  --private-label tls_ec_private \
  --private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000012000b",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],

```

```

    "cluster-coverage": "session"
  },
  "attributes": {
    "key-type": "ec",
    "label": "tls_ec_pub",
    "id": "",
    "check-value": "0xd7c1a7",
    "class": "public-key",
    "encrypt": false,
    "decrypt": false,
    "token": false,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 57,
    "ec-point":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3",
    "curve": "secp224r1"
  }
},
"private_key": {
  "key-reference": "0x000000000012000c",
  "key-info": {
    "key-owners": [
      {
        "username": "cu1",
        "key-coverage": "full"
      }
    ],
    "shared-users": [],
    "cluster-coverage": "session"
  }
},

```

```

    "attributes": {
      "key-type": "ec",
      "label": "tls_ec_private",
      "id": "",
      "check-value": "0xd7c1a7",
      "class": "private-key",
      "encrypt": false,
      "decrypt": false,
      "token": false,
      "always-sensitive": true,
      "derive": false,
      "destroyable": true,
      "extractable": true,
      "local": true,
      "modifiable": true,
      "never-extractable": false,
      "private": true,
      "sensitive": true,
      "sign": true,
      "trusted": false,
      "unwrap": false,
      "verify": false,
      "wrap": false,
      "wrap-with-trusted": false,
      "key-length-bytes": 122,
      "ec-point":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
      "curve": "secp224r1"
    }
  }
}
}

```

Exportar um arquivo de chave privada PEM falso

Depois de ter uma chave privada no HSM, você deve exportar um arquivo de chave privada PEM falso. Esse arquivo não contém os dados reais da chave, mas permite que o OpenSSL Dynamic Engine identifique a chave privada no HSM. Em seguida, use a chave privada para criar uma solicitação de assinatura de certificado (CSR) e assinar o CSR para criar um certificado.

Execute o comando [key generate-file](#) para exportar a chave privada no formato PEM falso e salvá-lo em um arquivo. Substitua os seguintes valores pelo seu próprio.

- `<private_key_label>`— Etiqueta da chave privada que você gerou na etapa anterior.
- `<web_server_fake_pem.key>`— Nome do arquivo no qual sua chave PEM falsa será gravada.

```
aws-cloudhsm > key generate-file --encoding reference-pem --  
path <web_server_fake_pem.key> --filter attr.label=<private_key_label>  
{  
  "error_code": 0,  
  "data": {  
    "message": "Successfully generated key file"  
  }  
}
```

Saia da CloudHSM CLI

Execute o comando a seguir para parar a CloudHSM CLI.

```
aws-cloudhsm > quit
```

Agora você deve ter um novo arquivo em seu sistema, localizado no caminho especificado `<web_server_fake_pem.key>` no comando anterior. Esse arquivo é o arquivo de chave privada PEM falso.

Gere um certificado autoassinado

Depois de gerar uma chave privada PEM falsa, você pode usar esse arquivo para gerar uma solicitação de assinatura de certificado (CSR) e um certificado.

Em um ambiente de produção, geralmente usa-se uma autoridade de certificação (CA) para criar um certificado de uma CSR. Não é necessária uma CA para um ambiente de teste. Se você usa uma CA, envie o arquivo CSR para eles e use o SSL/TLS certificado assinado que eles fornecem em seu servidor web para HTTPS.

Como alternativa ao uso de uma CA, você pode usar o AWS CloudHSM OpenSSL Dynamic Engine para criar um certificado autoassinado. Os certificados autoassinados não são confiáveis para os navegadores e não devem ser usados em ambientes de produção. Eles podem ser usados em ambientes de teste.

 Warning

Os certificados autoassinados devem ser usados apenas em um ambiente de teste. Para um ambiente de produção, use um método mais seguro, como uma autoridade de certificação, para criar um certificado.

Instalar e configurar o OpenSSL Dynamic Engine

1. Conecte-se à instância do cliente.
2. [the section called “Instalar”](#)

Gere um certificado

1. Obtenha uma cópia do arquivo PEM falso gerado em uma etapa anterior.
2. Crie uma CSR

Execute o comando a seguir para usar o AWS CloudHSM OpenSSL Dynamic Engine para criar uma solicitação de assinatura de certificado (CSR). `<web_server_fake_pem.key>` Substitua pelo nome do arquivo que contém sua chave privada PEM falsa. `<web_server.csr>` Substitua pelo nome do arquivo que contém sua CSR.

O comando `req` é interativo. Responda a cada campo. As informações do campo são copiadas em seu SSL/TLS certificado.

```
$ openssl req -engine cloudhsm -new -key <web_server_fake_pem.key> -  
out <web_server.csr>
```

3. Criar um certificado autoassinado

Execute o comando a seguir para usar o AWS CloudHSM OpenSSL Dynamic Engine para assinar sua CSR com sua chave privada em seu HSM. Isso cria um certificado autoassinado. Substitua os valores a seguir no comando pelos seus próprios.

- `<web_server.csr>`— Nome do arquivo que contém o CSR.
- `<web_server_fake_pem.key>`— Nome do arquivo que contém a chave privada PEM falsa.
- `<web_server.crt>`— Nome do arquivo que conterá seu certificado de servidor web.


```
$ openssl x509 -engine cloudhsm -req -days 365 -in <web_server.csr> -  
signkey <web_server_fake_pem.key> -out <web_server.crt>
```

Depois de concluir essas etapas, vá para [Etapa 3: configure o servidor Web](#).

Etapa 3: configure o servidor Web

Atualize a configuração do software de servidor web para usar o certificado HTTPS e a chave privada PEM falsa correspondente que você criou na [etapa anterior](#). Lembre-se de fazer backup de seus certificados e chaves existentes antes de começar. Isso concluirá a configuração do software do servidor web Linux para SSL/TLS descargado com AWS CloudHSM.

Conclua as etapas de uma das seções a seguir.

Tópicos

- [Configurar o servidor da Web NGINX](#)
- [Configure o servidor Web Apache](#)

Configurar o servidor da Web NGINX

Use esta seção para configurar o NGINX em plataformas compatíveis.

Para atualizar a configuração do servidor web para NGINX

1. Conecte-se à instância do cliente.
2. Execute o seguinte comando para criar os diretórios necessários para o certificado do servidor Web e a chave privada PEM falsa.

```
$ sudo mkdir -p /etc/pki/nginx/private
```

3. Execute o seguinte comando para copiar o certificado do seu servidor Web para o local desejado. `<web_server.crt>` Substitua pelo nome do certificado do seu servidor web.

```
$ sudo cp <web_server.crt> /etc/pki/nginx/server.crt
```

4. Execute o seguinte comando para copiar sua chave privada PEM falsa no local desejado.
`<web_server_fake_pem.key>` Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

```
$ sudo cp <web_server_example_pem.key> /etc/pki/nginx/private/server.key
```

5. Execute o comando a seguir para alterar a propriedade dos arquivos, para que o usuário chamado nginx possa lê-los.

```
$ sudo chown nginx /etc/pki/nginx/server.crt /etc/pki/nginx/private/server.key
```

6. Execute o comando a seguir para fazer backup do arquivo `/etc/nginx/nginx.conf`.

```
$ sudo cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.backup
```

7. Atualizar a configuração para NGINX.

Note

Cada cluster pode suportar no máximo 1000 processos de trabalho do NGINX em todos os servidores Web do NGINX.

Amazon Linux

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.  
server {  
    listen      443 ssl http2 default_server;  
    listen      [::]:443 ssl http2 default_server;  
    server_name _;  
    root        /usr/share/nginx/html;
```

```

ssl_certificate "/etc/pki/nginx/server.crt";
ssl_certificate_key "/etc/pki/nginx/private/server.key";
# It is *strongly* recommended to generate unique DH parameters
# Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
#ssl_dhparam "/etc/pki/nginx/dhparams.pem";
ssl_session_cache shared:SSL:1m;
ssl_session_timeout 10m;
ssl_protocols TLSv1.2;
ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
ssl_prefer_server_ciphers on;

# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {

error_page 404 /404.html;
location = /40x.html {

error_page 500 502 503 504 /50x.html;
location = /50x.html {

}
}

```

Amazon Linux 2

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

ssl_engine cloudhsm;
env CLOUDHSM_PIN;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is strongly recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Amazon Linux 2023

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.  
server {  
    listen      443 ssl http2 default_server;  
    listen      [::]:443 ssl http2 default_server;  
    server_name _;  
    root        /usr/share/nginx/html;  
  
    ssl_certificate "/etc/pki/nginx/server.crt";  
    ssl_certificate_key "/etc/pki/nginx/private/server.key";  
    # It is strongly recommended to generate unique DH parameters  
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048  
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";  
    ssl_session_cache shared:SSL:1m;  
    ssl_session_timeout 10m;  
    ssl_protocols TLSv1.2;  
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";  
    ssl_prefer_server_ciphers on;  
  
    # Load configuration files for the default server block.  
    include /etc/nginx/default.d/*.conf;  
  
    location / {  
    }  
  
    error_page 404 /404.html;  
    location = /40x.html {  
    }  
}
```

```
error_page 500 502 503 504 /50x.html;  
location = /50x.html {  
}  
}
```

CentOS 7

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.  
server {  
    listen      443 ssl http2 default_server;  
    listen      [::]:443 ssl http2 default_server;  
    server_name _;  
    root        /usr/share/nginx/html;  
  
    ssl_certificate "/etc/pki/nginx/server.crt";  
    ssl_certificate_key "/etc/pki/nginx/private/server.key";  
    # It is *strongly* recommended to generate unique DH parameters  
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048  
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";  
    ssl_session_cache shared:SSL:1m;  
    ssl_session_timeout 10m;  
    ssl_protocols TLSv1.2;  
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";  
    ssl_prefer_server_ciphers on;  
  
    # Load configuration files for the default server block.  
    include /etc/nginx/default.d/*.conf;
```

```

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}

```

CentOS 8

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

ssl_engine cloudhsm;
env CLOUDHSM_PIN;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```

# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is strongly recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-

```

```

SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {

    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}

```

Red Hat 7

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

ssl_engine cloudhsm;
env CLOUDHSM_PIN;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```

# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is strongly recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048

```



```
#ssl_dhparam "/etc/pki/nginx/dhparams.pem";
ssl_session_cache shared:SSL:1m;
ssl_session_timeout 10m;
ssl_protocols TLSv1.2;
ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
ssl_prefer_server_ciphers on;

# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {

error_page 404 /404.html;
location = /40x.html {

error_page 500 502 503 504 /50x.html;
location = /50x.html {

}
}
```

Red Hat 8

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
```

```

server_name _;
root        /usr/share/nginx/html;

ssl_certificate "/etc/pki/nginx/server.crt";
ssl_certificate_key "/etc/pki/nginx/private/server.key";
# It is strongly recommended to generate unique DH parameters
# Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
#ssl_dhparam "/etc/pki/nginx/dhparams.pem";
ssl_session_cache shared:SSL:1m;
ssl_session_timeout 10m;
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
ssl_prefer_server_ciphers on;

# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {

error_page 404 /404.html;
location = /40x.html {

error_page 500 502 503 504 /50x.html;
location = /50x.html {

}
}

```

Ubuntu 16.04 LTS

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

ssl_engine clouhsm;
env n3fips_password;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem
2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Ubuntu 18.04 LTS

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
ssl_engine cloudhsm;
    env CLOUDHSM_PIN;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem
2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
```

```

    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}

```

Ubuntu 20.04 LTS

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

ssl_engine cloudhsm;
env CLOUDHSM_PIN;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```

# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem
2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

```

```
# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {

}

error_page 404 /404.html;
location = /40x.html {
}

error_page 500 502 503 504 /50x.html;
location = /50x.html {
}
}
```

Ubuntu 22.04 LTS

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is strongly recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem
2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2 TLSv1.3;
```

```

    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
    SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-
    SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-
    RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-
    GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
    AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}

```

Ubuntu 24.04 LTS

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso requer permissões raiz do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

ssl_engine cloudhsm;
env CLOUDHSM_PIN;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```

# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";

```

```

ssl_certificate_key "/etc/pki/nginx/private/server.key";
# It is strongly recommended to generate unique DH parameters
# Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem
2048
#ssl_dhparam "/etc/pki/nginx/dhparams.pem";
ssl_session_cache shared:SSL:1m;
ssl_session_timeout 10m;
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-
SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-
RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-
GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
ssl_prefer_server_ciphers on;

# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {

error_page 404 /404.html;
location = /40x.html {

error_page 500 502 503 504 /50x.html;
location = /50x.html {

}
}

```

Salve o arquivo.

8. Faça o backup do arquivo de configuração systemd e defina o caminho EnvironmentFile.

Amazon Linux

Nenhuma ação necessária.

Amazon Linux 2

1. Faça backup do arquivo nginx.service.


```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção [Serviço], adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Amazon Linux 2023

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra `/lib/systemd/system/nginx.service` em um editor de texto. Na seção [Serviço], adicione:

```
EnvironmentFile=/etc/sysconfig/nginx
```

CentOS 7

Nenhuma ação necessária.

CentOS 8

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção [Serviço], adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Red Hat 7

Nenhuma ação necessária.

Red Hat 8

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção [Serviço], adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 16.04

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção [Serviço], adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 18.04

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção [Serviço], adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 20.04 LTS

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção `[Serviço]`, adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 22.04 LTS

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção `[Serviço]`, adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 24.04 LTS

1. Faça backup do arquivo `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra o arquivo `/lib/systemd/system/nginx.service` em um editor de texto e, na seção `[Serviço]`, adicione o seguinte caminho:

```
EnvironmentFile=/etc/sysconfig/nginx
```

9. Verifique se o arquivo `/etc/sysconfig/nginx` existe e siga um destes procedimentos:

- Se o arquivo existir, faça backup do arquivo executando o seguinte comando:

```
$ sudo cp /etc/sysconfig/nginx /etc/sysconfig/nginx.backup
```

- Se o arquivo não existir, abra um editor de texto e crie um arquivo chamado `nginx` na pasta `/etc/sysconfig/`.

10. Configure o ambiente NGINX.

Note

O Client SDK 5 introduz a variável de ambiente `CLOUDHSM_PIN` para armazenar as credenciais do CU.

Amazon Linux

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Amazon Linux 2

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Amazon Linux 2023

Como usuário-raiz do Linux, abra o arquivo `/etc/sysconfig/nginx` em um editor de texto. Por exemplo,

```
sudo vi /etc/sysconfig/nginx
```

Adicione as credenciais do Cryptography User (CU):

```
CLLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

CentOS 7

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

CentOS 8

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
CLLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Red Hat 7

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Red Hat 8

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Ubuntu 16.04 LTS

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
n3fips_password=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Ubuntu 18.04 LTS

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Ubuntu 20.04 LTS

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Ubuntu 22.04 LTS

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

Ubuntu 24.04 LTS

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Salve o arquivo.

11. Inicie o servidor web NGINX.

Amazon Linux

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso requer permissões raiz do Linux. Adicione as credenciais do Cryptography User (CU):

```
$ sudo service nginx start
```

Amazon Linux 2

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Amazon Linux 2023

Parar todos os processos do NGINX

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Iniciar o NGINX

```
$ sudo systemctl start nginx
```

CentOS 7

Parar qualquer processo NGINX em execução


```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

CentOS 8

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Red Hat 7

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Red Hat 8

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 16.04 LTS

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 18.04 LTS

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 20.04 LTS

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 22.04 LTS

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 24.04 LTS

Pare qualquer processo NGINX em execução

```
$ sudo systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ sudo systemctl daemon-reload
```

Inicie o processo NGINX

```
$ sudo systemctl start nginx
```

12. (Opcional) Configure sua plataforma para iniciar o NGINX na inicialização.

Amazon Linux

```
$ sudo chkconfig nginx on
```

Amazon Linux 2

```
$ sudo systemctl enable nginx
```

Amazon Linux 2023

```
$ sudo systemctl enable nginx
```

CentOS 7

Nenhuma ação necessária.

CentOS 8

```
$ sudo systemctl enable nginx
```

Red Hat 7

Nenhuma ação necessária.

Red Hat 8

```
$ sudo systemctl enable nginx
```

Ubuntu 16.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 18.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 20.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 22.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 24.04 LTS

```
$ sudo systemctl enable nginx
```

Depois de atualizar a configuração do servidor web, vá para [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#).

Configure o servidor Web Apache

Use esta seção para configurar o Apache em plataformas compatíveis.

Para atualizar a configuração do servidor Web para o Apache

1. Conecte-se à sua instância do cliente Amazon EC2.
2. Defina locais padrão para certificados e chaves privadas para sua plataforma.

Amazon Linux

No arquivo `/etc/httpd/conf.d/ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

Amazon Linux 2

No arquivo `/etc/httpd/conf.d/ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

Amazon Linux 2023

Abra o arquivo `/etc/httpd/conf.d/ssl.conf`. Adicione estes valores se eles ainda não existirem:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

CentOS 7

No arquivo `/etc/httpd/conf.d/ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

CentOS 8

No arquivo `/etc/httpd/conf.d/ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

Red Hat 7

No arquivo `/etc/httpd/conf.d/ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

Red Hat 8

No arquivo `/etc/httpd/conf.d/ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/pki/tls/certs/localhost.crt  
SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
```

Ubuntu 16.04 LTS

No arquivo `/etc/apache2/sites-available/default-ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt  
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

Ubuntu 18.04 LTS

No arquivo `/etc/apache2/sites-available/default-ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt  
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

Ubuntu 20.04 LTS

No arquivo `/etc/apache2/sites-available/default-ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt  
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

Ubuntu 22.04 LTS

No arquivo `/etc/apache2/sites-available/default-ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt  
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

Ubuntu 24.04 LTS

No arquivo `/etc/apache2/sites-available/default-ssl.conf`, certifique-se de que esses valores existam:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt  
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

3. Copie o certificado do seu servidor web para o local necessário para sua plataforma.

Amazon Linux

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt> Substitua pelo nome do certificado do seu servidor web.

Amazon Linux 2

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt> Substitua pelo nome do certificado do seu servidor web.

Amazon Linux 2023

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt> Substitua pelo nome do certificado do seu servidor web.

CentOS 7

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt> Substitua pelo nome do certificado do seu servidor web.

CentOS 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt> Substitua pelo nome do certificado do seu servidor web.

Red Hat 7

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt> Substitua pelo nome do certificado do seu servidor web.

Red Hat 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

<web_server.crt>Substitua pelo nome do certificado do seu servidor web.

Ubuntu 16.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

<web_server.crt>Substitua pelo nome do certificado do seu servidor web.

Ubuntu 18.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

<web_server.crt>Substitua pelo nome do certificado do seu servidor web.

Ubuntu 20.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

<web_server.crt>Substitua pelo nome do certificado do seu servidor web.

Ubuntu 22.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

<web_server.crt>Substitua pelo nome do certificado do seu servidor web.

Ubuntu 24.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

<web_server.crt>Substitua pelo nome do certificado do seu servidor web.

4. Copie sua chave privada PEM falsa para o local necessário para sua plataforma.

Amazon Linux

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Amazon Linux 2

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Amazon Linux 2023

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

CentOS 7

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

CentOS 8

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Red Hat 7

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Red Hat 8

```
$ sudo cp <web_server_example_pem.key> /etc/pki/tls/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Ubuntu 16.04 LTS

```
$ sudo cp <web_server_example_pem.key> /etc/ssl/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Ubuntu 18.04 LTS

```
$ sudo cp <web_server_example_pem.key> /etc/ssl/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Ubuntu 20.04 LTS

```
$ sudo cp <web_server_example_pem.key> /etc/ssl/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Ubuntu 22.04 LTS

```
$ sudo cp <web_server_example_pem.key> /etc/ssl/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

Ubuntu 24.04 LTS

```
$ sudo cp <web_server_example_pem.key> /etc/ssl/private/localhost.key
```

<web_server_example_pem.key> Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

5. Altere a propriedade desses arquivos, se exigido pela sua plataforma.

Amazon Linux

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

Amazon Linux 2

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

Amazon Linux 2023

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

CentOS 7

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

CentOS 8

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

Red Hat 7

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

Red Hat 8

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/localhost.key
```

Forneça permissão de leitura para o usuário chamado apache.

Ubuntu 16.04 LTS

Nenhuma ação necessária.

Ubuntu 18.04 LTS

Nenhuma ação necessária.

Ubuntu 20.04 LTS

Nenhuma ação necessária.

Ubuntu 22.04 LTS

Nenhuma ação necessária.

Ubuntu 24.04 LTS

Nenhuma ação necessária.

6. Configure as diretivas do Apache para sua plataforma.

Amazon Linux

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cLoudhsm  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

Salve o arquivo.

Amazon Linux 2

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cLoudhsm  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

Salve o arquivo.

Amazon Linux 2023

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

O arquivo de configuração do Apache define o comportamento do servidor. Edite esse arquivo com permissões de raiz.

Atualize ou adicione as seguintes diretivas:

```
SSLCryptoDevice cloudhsm  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

Salve o arquivo.

CentOS 7

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cloudhsm  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

Salve o arquivo.

CentOS 8

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice ccloudhsm
SSLProtocol TLSv1.2 TLSv1.3
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProxyCipherSuite HIGH:!aNULL
```

Salve o arquivo.

Red Hat 7

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice ccloudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```


Salve o arquivo.

Red Hat 8

Localize o arquivo SSL dessa plataforma:

```
/etc/httpd/conf.d/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cloudhsm  
SSLProtocol TLSv1.2 TLSv1.3  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA  
SSLProxyCipherSuite HIGH:!aNULL
```

Salve o arquivo.

Ubuntu 16.04 LTS

Localize o arquivo SSL dessa plataforma:

```
/etc/apache2/mods-available/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cloudhsm  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
```

```
RSA-AES128-SHA256:ECDSA-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDSA-ECDSA-AES256-GCM-SHA384:ECDSA-ECDSA-AES256-SHA384:ECDSA-ECDSA-AES128-GCM-SHA256:ECDSA-ECDSA-AES128-SHA256:ECDSA-ECDSA-AES256-SHA:ECDSA-ECDSA-AES128-SHA
```

Salve o arquivo.

Ative o módulo SSL e a configuração padrão do site SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 18.04 LTS

Localize o arquivo SSL dessa plataforma:

```
/etc/apache2/mods-available/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cloudhsm
SSLCipherSuite ECDSA-RSA-AES128-GCM-SHA256:ECDSA-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDSA-RSA-AES256-SHA384:ECDSA-RSA-AES128-SHA256:ECDSA-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDSA-ECDSA-AES256-GCM-SHA384:ECDSA-ECDSA-AES256-SHA384:ECDSA-ECDSA-AES128-GCM-SHA256:ECDSA-ECDSA-AES128-SHA256:ECDSA-ECDSA-AES256-SHA:ECDSA-ECDSA-AES128-SHA
SSLProtocol TLSv1.2 TLSv1.3
```

Salve o arquivo.

Ative o módulo SSL e a configuração padrão do site SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 20.04 LTS

Localize o arquivo SSL dessa plataforma:

```
/etc/apache2/mods-available/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```
SSLCryptoDevice cloudhsm  
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA  
SSLProtocol TLSv1.2 TLSv1.3
```

Salve o arquivo.

Ative o módulo SSL e a configuração padrão do site SSL:

```
$ sudo a2enmod ssl  
$ sudo a2ensite default-ssl
```

Ubuntu 22.04 LTS

Localize o arquivo SSL dessa plataforma:

```
/etc/apache2/mods-available/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```

SSLCryptoDevice cLoudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProtocol TLSv1.2 TLSv1.3

```

Salve o arquivo.

Ative o módulo SSL e a configuração padrão do site SSL:

```

$ sudo a2enmod ssl
$ sudo a2ensite default-ssl

```

Ubuntu 24.04 LTS

Localize o arquivo SSL dessa plataforma:

```
/etc/apache2/mods-available/ssl.conf
```

Esse arquivo contém diretivas do Apache que definem como seu servidor deve ser executado. As diretivas aparecem à esquerda, seguidas por um valor. Use um editor de texto para editar esse arquivo. Isso requer permissões raiz do Linux.

Atualize ou insira as seguintes diretivas com esses valores:

```

SSLCryptoDevice cLoudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProtocol TLSv1.2 TLSv1.3

```

Salve o arquivo.

Ative o módulo SSL e a configuração padrão do site SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

7. Configure um arquivo de valores de ambiente para sua plataforma.

Amazon Linux

Nenhuma ação necessária. Os valores ambientais entram em `/etc/sysconfig/httpd`

Amazon Linux 2

Abra o arquivo do serviço httpd:

```
/lib/systemd/system/httpd.service
```

Adicione o seguinte à seção do `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Amazon Linux 2023

Abra o `/lib/systemd/system/httpd.service`

Na seção `[Serviço]`, adicione:

```
EnvironmentFile=/etc/sysconfig/httpd
```

CentOS 7

Abra o arquivo do serviço httpd:

```
/lib/systemd/system/httpd.service
```

Adicione o seguinte à seção do `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

CentOS 8

Abra o arquivo do serviço httpd:

```
/lib/systemd/system/httpd.service
```

Adicione o seguinte à seção do [Service]:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Red Hat 7

Abra o arquivo do serviço httpd:

```
/lib/systemd/system/httpd.service
```

Adicione o seguinte à seção do [Service]:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Red Hat 8

Abra o arquivo do serviço httpd:

```
/lib/systemd/system/httpd.service
```

Adicione o seguinte à seção do [Service]:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Ubuntu 16.04 LTS

Nenhuma ação necessária. Os valores ambientais entram em /etc/sysconfig/httpd

Ubuntu 18.04 LTS

Nenhuma ação necessária. Os valores ambientais entram em /etc/sysconfig/httpd

Ubuntu 20.04 LTS

Nenhuma ação necessária. Os valores ambientais entram em /etc/sysconfig/httpd

Ubuntu 22.04 LTS

Nenhuma ação necessária. Os valores ambientais entram em `/etc/sysconfig/httpd`

Ubuntu 24.04 LTS

Nenhuma ação necessária. Os valores ambientais entram em `/etc/sysconfig/httpd`

8. No arquivo que armazena variáveis de ambiente para sua plataforma, defina uma variável de ambiente que contenha as credenciais do usuário criptográfico (CU):

Amazon Linux

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua `<CU user name>` e `<password>` pelas credenciais da UC.

Amazon Linux 2

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua `<CU user name>` e `<password>` pelas credenciais da UC.

Amazon Linux 2023

Abra `/etc/sysconfig/httpd`, adicione:

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua `<CU user name>` e `<password>` pelas credenciais da UC.

CentOS 7

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

CentOS 8

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Red Hat 7

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

```
ssl_engine cloudhsm;  
env CLOUDHSM_PIN;
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Red Hat 8

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Note

O Client SDK 5 introduz a variável de ambiente CLOUDHSM_PIN para armazenar as credenciais do CU.

Ubuntu 16.04 LTS

Use um editor de texto para editar o `/etc/apache2/envvars`.

```
export n3fips_password=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

Ubuntu 18.04 LTS

Use um editor de texto para editar o `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua `<CU user name>` e `<password>` pelas credenciais da UC.

Note

O Client SDK 5 introduz a variável de ambiente CLOUDHSM_PIN para armazenar as credenciais do CU. No SDK do cliente 3, você armazenou as credenciais da UC na variável de ambiente `n3fips_password`. O Client SDK 5 é compatível com as duas variáveis de ambiente, mas recomendamos o uso de CLOUDHSM_PIN.

Ubuntu 20.04 LTS

Use um editor de texto para editar o `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua `<CU user name>` e `<password>` pelas credenciais da UC.

Note

O Client SDK 5 introduz a variável de ambiente CLOUDHSM_PIN para armazenar as credenciais do CU. No SDK do cliente 3, você armazenou as credenciais da UC na variável de ambiente `n3fips_password`. O Client SDK 5 é compatível com as duas variáveis de ambiente, mas recomendamos o uso de CLOUDHSM_PIN.

Ubuntu 22.04 LTS

Use um editor de texto para editar o `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

 Note

O Client SDK 5 introduz a variável de ambiente CLOUDHSM_PIN para armazenar as credenciais do CU. No SDK do cliente 3, você armazenou as credenciais da UC na variável de ambiente n3fips_password. O Client SDK 5 é compatível com as duas variáveis de ambiente, mas recomendamos o uso de CLOUDHSM_PIN.

Ubuntu 24.04 LTS

Use um editor de texto para editar o `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC.

 Note

O Client SDK 5 introduz a variável de ambiente CLOUDHSM_PIN para armazenar as credenciais do CU. No SDK do cliente 3, você armazenou as credenciais da UC na variável de ambiente n3fips_password. O Client SDK 5 é compatível com as duas variáveis de ambiente, mas recomendamos o uso de CLOUDHSM_PIN.

9. Inicie o servidor web Apache.

Amazon Linux

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

Amazon Linux 2

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

Amazon Linux 2023

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

CentOS 7

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

CentOS 8

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

Red Hat 7

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

Red Hat 8

```
$ sudo systemctl daemon-reload  
$ sudo service httpd start
```

Ubuntu 16.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 18.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 20.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 22.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 24.04 LTS

```
$ sudo service apache2 start
```

10. (Opcional) Configure sua plataforma para iniciar o Apache na inicialização.

Amazon Linux

```
$ sudo chkconfig httpd on
```

Amazon Linux 2

```
$ sudo chkconfig httpd on
```

Amazon Linux 2023

```
$ sudo chkconfig httpd on
```

CentOS 7

```
$ sudo chkconfig httpd on
```

CentOS 8

```
$ systemctl enable httpd
```

Red Hat 7

```
$ sudo chkconfig httpd on
```

Red Hat 8

```
$ systemctl enable httpd
```

Ubuntu 16.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 18.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 20.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 22.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 24.04 LTS

```
$ sudo systemctl enable apache2
```

Depois de atualizar a configuração do servidor web, vá para [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#).

Etapa 4: permitir tráfego HTTPS e verificar o certificado

Depois de configurar seu servidor web para SSL/TLS descargando AWS CloudHSM, adicione sua instância de servidor web a um grupo de segurança que permite tráfego HTTPS de entrada. Isso permite que clientes, como navegadores da Web, estabeleçam uma conexão HTTPS com seu servidor Web. Em seguida, faça uma conexão HTTPS com seu servidor web e verifique se ele está usando o certificado que você configurou para SSL/TLS descargando AWS CloudHSM.

Tópicos

- [Habilitar conexões HTTPS de entrada](#)
- [Verificar se o HTTPS usa o certificado que você configurou](#)

Habilitar conexões HTTPS de entrada

Para se conectar ao seu servidor Web a partir de um cliente (como um navegador da Web), crie um grupo de segurança que permita conexões HTTPS de entrada. Especificamente, ele deve permitir conexões TCP de entrada na porta 443. Atribua esse grupo de segurança ao seu servidor Web.

Para criar um grupo de segurança para HTTPS e atribuí-lo ao seu servidor Web

1. Abra o console do Amazon EC2 em <https://console.aws.amazon.com/ec2/>.
2. No painel de navegação, escolha Grupos de segurança.
3. Escolha Create security group (Criar grupo de segurança).
4. Para Create Security Group, faça o seguinte:
 - a. Para Security group name, digite um nome para security group que você está criando.
 - b. (Opcional) Digite uma descrição do security group que você está criando.
 - c. Para VPC, escolha a VPC que contém a instância do Amazon EC2 do servidor Web.
 - d. Selecione Adicionar regra.
 - e. Em Tipo, selecione HTTPS na janela suspensa.
 - f. Em Fonte, insira um local de origem.
 - g. Escolha Create security group (Criar grupo de segurança).
5. No painel de navegação, escolha Instâncias.
6. Marque a caixa de seleção ao lado da sua instância do servidor Web.
7. Selecione o menu suspenso Ações, na parte superior da página. Selecione Segurança e, em seguida, Alterar grupos de segurança.
8. Em Grupos de segurança associados, escolha a caixa de pesquisa e escolha o grupo de segurança que você criou para HTTPS. Em seguida, escolha Adicionar grupos de segurança.
9. Selecione Save (Salvar).

Verificar se o HTTPS usa o certificado que você configurou

Depois de adicionar o servidor web a um grupo de segurança, você pode verificar se o SSL/TLS offload está usando seu certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como [OpenSSL s_client](#).

Para verificar o SSL/TLS descarregamento com um navegador da web

1. Use um navegador da Web para se conectar ao servidor Web usando o nome de DNS público ou endereço IP do servidor. Certifique-se de que a URL na barra de endereços comece com `https://`. Por exemplo, `https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, `https://www.example.com/`) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Use seu navegador da Web para ver o certificado do servidor Web. Para obter mais informações, consulte:
 - Para o Mozilla Firefox, consulte o tópico sobre como [Visualizar um certificado](#), no site de suporte da Mozilla.
 - Para o Google Chrome, consulte [Noções básicas de problemas de segurança](#), no site Google Tools for Web Developers.

Outros navegadores da Web podem ter recursos semelhantes, que você pode usar para visualizar o certificado do servidor Web.

3. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Para verificar o SSL/TLS descarregamento com o OpenSSL `s_client`

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS.
`<server name>` Substitua pelo nome DNS público ou endereço IP do seu servidor web.

```
openssl s_client -connect <server name>:443
```

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Agora você tem um site protegido com HTTPS. A chave privada do servidor web é armazenada em um HSM no seu AWS CloudHSM cluster.

Para adicionar um balanceador de carga, consulte [Adicione um balanceador de carga com o Elastic Load Balancing AWS CloudHSM para \(opcional\)](#).

AWS CloudHSM Descarregamento de SSL/TLS no Linux usando NGINX ou com o OpenSSL Provider HAProxy

Este tópico fornece step-by-step instruções para configurar o descarregamento de identidade SSL/TLS do servidor AWS CloudHSM em um servidor web Linux usando o NGINX ou com HAProxy o OpenSSL Provider.

Tópicos

- [Visão geral do](#)
- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: gerar ou importar uma chave privada e obter um certificado](#)
- [Etapa 3: configure o servidor Web](#)
- [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#)

Visão geral do

No Linux, o [NGINX](#) e o software do servidor [HAProxy](#)web se integram ao [OpenSSL](#) para oferecer suporte a HTTPS. O [AWS CloudHSM OpenSSL](#) Provider fornece uma interface que permite que o software do servidor web use HSMs o em seu cluster para descarga criptográfica e armazenamento

de chaves. O OpenSSL Provider é a ponte que conecta o servidor web ao seu cluster. AWS CloudHSM

Para concluir este tutorial, você configurará o NGINX ou HAProxy usará o OpenSSL Provider AWS CloudHSM . O tutorial mostra como fazer o seguinte:

- Instale o software do servidor Web em uma instância do Amazon EC2.
- Configure o software do servidor Web para oferecer suporte a HTTPS com uma chave privada armazenada em seu AWS CloudHSM cluster.
- (Opcional) Use o Amazon EC2 para criar uma segunda instância de servidor Web e o Elastic Load Balancing para criar um balanceador de carga. Usar um load balanceador de carga pode aumentar o desempenho, distribuindo a carga em vários servidores. Ele também pode fornecer redundância e maior disponibilidade se um ou mais servidores falhar.

Quando estiver pronto para começar, vá para [Etapa 1: configurar os pré-requisitos](#).

Etapa 1: configurar os pré-requisitos

Plataformas diferentes exigem pré-requisitos diferentes. Use a seção de pré-requisitos abaixo que corresponde à sua plataforma.

Pré-requisitos para o OpenSSL Provider AWS CloudHSM

Para configurar o descarregamento de identidade SSL/TLS do servidor web com o AWS CloudHSM OpenSSL Provider for Client SDK 5, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos dois módulos de segurança de hardware (HSM)

Note

Você pode usar um único cluster HSM, mas primeiro deve desativar a durabilidade da chave do cliente. Para obter mais informações, consulte [Gerenciar configurações de durabilidade da chave do cliente](#) e [Ferramenta de configuração do Client SDK 5](#).

- Uma instância do Amazon EC2 executando um sistema operacional Linux com o seguinte software instalado:
 - Um servidor web (NGINX ou) HAProxy
 - O provedor de AWS CloudHSM OpenSSL para Client SDK 5

- Um [usuário de criptografia](#) (CU) para ter e gerenciar a chave privada do servidor Web no HSM.

Para configurar uma instância do servidor web do Linux e criar um CU no HSM

 Note

Muitos dos comandos desse procedimento exigem privilégios elevados. Talvez seja necessário executar comandos com sudo ou como usuário root, dependendo da configuração do sistema.

1. Instale e configure o AWS CloudHSM OpenSSL Provider for Client SDK 5. Para obter mais informações sobre a instalação do OpenSSL Provider, consulte [AWS CloudHSM OpenSSL Provider for Client SDK 5](#).
2. Em uma instância do EC2 Linux que tenha acesso ao seu cluster, instale o NGINX ou HAProxy o servidor web:

Amazon Linux 2023

- NGINX

```
$ yum install nginx
```

- HAProxy

```
$ yum install haproxy
```

RHEL 9 (9.2+)

- NGINX

```
$ yum install nginx
```

- HAProxy

```
$ yum install haproxy
```

RHEL 10 (10.0+)

- NGINX

```
$ yum install nginx
```

- HAProxy

```
$ yum install haproxy
```

Ubuntu 24.04

- NGINX

```
$ apt install nginx
```

- HAProxy

```
$ apt install haproxy
```

3. Use a CloudHSM CLI para criar um [usuário de criptografia](#). Para obter mais informações sobre o gerenciamento de usuários do HSM, consulte [Gerenciar usuários do HSM com a CLI do CloudHSM](#).

Tip

Lembre o nome do usuário e a senha do CU. Eles serão necessários mais tarde ao gerar ou importar a chave privada HTTPS e o certificado para o servidor Web.

Depois de concluir essas etapas, vá para [Etapa 2: gerar ou importar uma chave privada e obter um certificado](#).

Observações

- Para usar o Security-Enhanced Linux (SELinux) e servidores web, você deve permitir conexões TCP de saída na porta 2223, que é a porta que o Client SDK 5 usa para se comunicar com o HSM.

- Para criar e ativar um cluster e dar acesso a uma instância do EC2 ao cluster, conclua as etapas em [Conceitos básicos de AWS CloudHSM](#). Os conceitos básicos oferecem step-by-step instruções para criar um cluster ativo com um HSM e uma instância cliente do Amazon EC2. Você pode usar essa instância de cliente como seu servidor Web.
- Para evitar a desativação da durabilidade da chave do cliente, adicione mais de um HSM ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
- Você pode usar um cliente SSH ou PuTTY para se conectar à instância do cliente. Para obter mais informações, consulte [Conectar à instância Linux utilizando SSH](#) ou [Conectar à instância Linux no Windows utilizando PuTTY](#) na documentação do Amazon C2.

Etapa 2: gerar ou importar uma chave privada e obter um certificado

Para habilitar o HTTPS, seu aplicativo de servidor web (NGINX ou HAProxy) precisa de uma chave privada e um certificado correspondente SSL/TLS. Para usar a identidade do SSL/TLS servidor web offload with AWS CloudHSM, você deve armazenar a chave privada em um HSM no seu cluster. AWS CloudHSM Crie uma chave privada e use a chave para criar uma solicitação de assinatura de certificado (CSR). Você exporta uma chave privada PEM falsa do HSM, que é um arquivo de chave privada no formato PEM que contém uma referência à chave privada armazenada no HSM (não é a chave privada real). Seu servidor web usa o arquivo de chave privada PEM falso para identificar a chave privada no HSM durante o descarregamento da identidade SSL/TLS do servidor.

Gerar uma chave privada

Esta seção mostra como gerar um par de chaves usando o [CloudHSM CLI](#). Depois de gerar um par de chaves dentro do HSM, você pode exportá-lo como um arquivo PEM falso e gerar o certificado correspondente.

Instalar e configurar a CloudHSM CLI

1. [Instale e configure](#) a CloudHSM CLI.
2. Use o comando a seguir para iniciar a CloudHSM CLI.

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

3. Execute o seguinte comando para fazer login no HSM. *<user name>* Substitua pelo nome de usuário do seu usuário criptográfico

```
aws-cloudhsm>login --username <user name> --role crypto-user
```

Gerar uma chave privada

Dependendo do seu caso de uso, você pode gerar um RSA ou um par de chaves EC. Execute um destes procedimentos:

- Para gerar uma chave privada RSA em um HSM

Use o comando [key generate-asymmetric-pair rsa](#) para gerar um par de chaves RSA. Este exemplo gera um par de chaves RSA com um módulo de 2048, um expoente público de 65537, rótulo de chave pública de *tls_rsa_pub* e rótulo de chave privada de *tls_rsa_private*.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label tls_rsa_pub \
--private-label tls_rsa_private \
--private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "tls_rsa_pub",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
```

```

    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 512,
    "public-exponent": "0x010001",
    "modulus":
"0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
    "modulus-size-bits": 2048
  },
  "private_key": {
    "key-reference": "0x00000000000280cc7",
    "key-info": {
      "key-owners": [
        {
          "username": "cu1",
          "key-coverage": "full"
        }
      ],
      "shared-users": [],
      "cluster-coverage": "full"
    },
    "attributes": {
      "key-type": "rsa",
      "label": "tls_rsa_private",
      "id": "",
      "check-value": "0x01fe6e",
      "class": "private-key",
      "encrypt": false,
      "decrypt": true,
      "token": true,
      "always-sensitive": true,
      "derive": false,

```

```

        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
"0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
        "modulus-size-bits": 2048
    }
}
}
}

```

- Para gerar uma chave privada EC em um HSM

Use o comando [key generate-asymmetric-pair ec](#) para gerar um par de chaves EC. Este exemplo gera um par de chaves EC com a prime256v1 curva (correspondente à NID_X9_62_prime256v1 curva), um rótulo de chave pública de *tls_ec_pub* e um rótulo de chave privada de *tls_ec_private*.

```

aws-cloudhsm > key generate-asymmetric-pair ec \
  --curve prime256v1 \
  --public-label tls_ec_pub \
  --private-label tls_ec_private \
  --private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000012000b",
      "key-info": {
        "key-owners": [
          {

```

```

        "username": "cu1",
        "key-coverage": "full"
    }
],
"shared-users": [],
"cluster-coverage": "session"
},
"attributes": {
    "key-type": "ec",
    "label": "tls_ec_pub",
    "id": "",
    "check-value": "0xd7c1a7",
    "class": "public-key",
    "encrypt": false,
    "decrypt": false,
    "token": false,
    "always-sensitive": false,
    "derive": false,
    "destroyable": true,
    "extractable": true,
    "local": true,
    "modifiable": true,
    "never-extractable": false,
    "private": true,
    "sensitive": false,
    "sign": false,
    "trusted": false,
    "unwrap": false,
    "verify": false,
    "wrap": false,
    "wrap-with-trusted": false,
    "key-length-bytes": 57,
    "ec-point":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
    "curve": "secp224r1"
}
},
"private_key": {
    "key-reference": "0x0000000000012000c",
    "key-info": {
        "key-owners": [
            {
                "username": "cu1",
                "key-coverage": "full"
            }
        ]
    }
}

```



```

    }
  ],
  "shared-users": [],
  "cluster-coverage": "session"
},
"attributes": {
  "key-type": "ec",
  "label": "tls_ec_private",
  "id": "",
  "check-value": "0xd7c1a7",
  "class": "private-key",
  "encrypt": false,
  "decrypt": false,
  "token": false,
  "always-sensitive": true,
  "derive": false,
  "destroyable": true,
  "extractable": true,
  "local": true,
  "modifiable": true,
  "never-extractable": false,
  "private": true,
  "sensitive": true,
  "sign": true,
  "trusted": false,
  "unwrap": false,
  "verify": false,
  "wrap": false,
  "wrap-with-trusted": false,
  "key-length-bytes": 122,
  "ec-point":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
  "curve": "secp224r1"
}
}
}
}

```

Exportar um arquivo de chave privada PEM falso

Depois de ter uma chave privada no HSM, você deve exportar um arquivo de chave privada PEM falso. Esse arquivo não contém os dados reais da chave, mas permite que o OpenSSL Dynamic

Engine identifique a chave privada no HSM. Em seguida, use a chave privada para criar uma solicitação de assinatura de certificado (CSR) e assinar o CSR para criar um certificado.

Execute o comando [key generate-file](#) para exportar a chave privada no formato PEM falso e salvá-lo em um arquivo. Substitua os seguintes valores pelo seu próprio.

- *<private_key_label>*— Etiqueta da chave privada que você gerou na etapa anterior.
- *<web_server_fake_pem.key>*— Nome do arquivo no qual sua chave PEM falsa será gravada.

```
aws-cloudhsm > key generate-file --encoding reference-pem --  
path <web_server_fake_pem.key> --filter attr.label=<private_key_label>  
{  
  "error_code": 0,  
  "data": {  
    "message": "Successfully generated key file"  
  }  
}
```

Saia da CloudHSM CLI

Execute o comando a seguir para parar a CloudHSM CLI.

```
aws-cloudhsm > quit
```

Agora você deve ter um novo arquivo em seu sistema, localizado no caminho especificado *<web_server_fake_pem.key>* no comando anterior. Esse arquivo é o arquivo de chave privada PEM falso.

Gere um certificado autoassinado

Depois de gerar uma chave privada PEM falsa, você pode usar esse arquivo para gerar uma solicitação de assinatura de certificado (CSR) e um certificado.

Em um ambiente de produção, geralmente usa-se uma autoridade de certificação (CA) para criar um certificado de uma CSR. Não é necessária uma CA para um ambiente de teste. Se você usa uma CA, envie o arquivo CSR para eles e use o SSL/TLS certificado assinado que eles fornecem em seu servidor web para HTTPS.

Como alternativa ao uso de uma CA, você pode usar o AWS CloudHSM OpenSSL Dynamic Engine para criar um certificado autoassinado. Os certificados autoassinados não são confiáveis para os

navegadores e não devem ser usados em ambientes de produção. Eles podem ser usados em ambientes de teste.

 Warning

Os certificados autoassinados devem ser usados apenas em um ambiente de teste. Para um ambiente de produção, use um método mais seguro, como uma autoridade de certificação, para criar um certificado.

Instalar e configurar o OpenSSL Dynamic Engine

1. Conecte-se à instância do cliente.
2. [the section called “Instalar”](#)

Gere um certificado

1. Obtenha uma cópia do arquivo PEM falso gerado em uma etapa anterior.
2. Crie uma CSR

Execute o comando a seguir para usar o AWS CloudHSM OpenSSL Dynamic Engine para criar uma solicitação de assinatura de certificado (CSR). `<web_server_fake_pem.key>` Substitua pelo nome do arquivo que contém sua chave privada PEM falsa. `<web_server.csr>` Substitua pelo nome do arquivo que contém sua CSR.

O comando `req` é interativo. Responda a cada campo. As informações do campo são copiadas em seu SSL/TLS certificado.

 Note

Atualmente, a criação de CSR não é suportada pelo OpenSSL Provider. Você deve usar o OpenSSL Engine para essa etapa, mas as operações de criptografia TLS funcionarão com o Provider.

```
$ openssl req -engine cloudhsm -new -key <web_server_fake_pem.key> -  
out <web_server.csr>
```

3. Criar um certificado autoassinado

Execute o comando a seguir para usar o AWS CloudHSM OpenSSL Dynamic Engine para assinar sua CSR com sua chave privada em seu HSM. Isso cria um certificado autoassinado. Substitua os valores a seguir no comando pelos seus próprios.

- `<web_server.csr>`— Nome do arquivo que contém o CSR.
- `<web_server_fake_pem.key>`— Nome do arquivo que contém a chave privada PEM falsa.
- `<web_server.crt>`— Nome do arquivo que conterá seu certificado de servidor web.

```
$ openssl x509 -engine cloudhsm -req -days 365 -in <web_server.csr> -  
signkey <web_server_fake_pem.key> -out <web_server.crt>
```

Depois de ter uma chave privada e um certificado, acesse [Etapa 3: configure o servidor Web](#).

Etapa 3: configure o servidor Web

Atualize a configuração do software de servidor web para usar o certificado HTTPS e a chave privada PEM falsa correspondente que você criou na [etapa anterior](#). Lembre-se de fazer backup de seus certificados e chaves existentes antes de começar. Isso concluirá a configuração do software de servidor web Linux para transferência da identidade do SSL/TLS servidor com AWS CloudHSM

Conclua as etapas de uma das seções a seguir.

Tópicos

- [Configurar o servidor da Web NGINX](#)
- [Configurar servidor HAProxy web](#)

Configurar o servidor da Web NGINX

Use esta seção para configurar o NGINX com o OpenSSL Provider.

Para configurar o NGINX para o OpenSSL Provider

1. Conecte-se à instância do cliente.
2. Execute o seguinte comando para criar os diretórios necessários para o certificado do servidor Web e a chave privada PEM falsa.

```
$ mkdir -p /etc/pki/nginx/private
```

3. Execute o seguinte comando para copiar o certificado do seu servidor Web para o local desejado. `<web_server.crt>` Substitua pelo nome do certificado do seu servidor web.

```
$ cp <web_server.crt> /etc/pki/nginx/server.crt
```

4. Execute o seguinte comando para copiar sua chave privada PEM falsa no local desejado. `<web_server_fake_pem.key>` Substitua pelo nome do arquivo que contém sua chave privada PEM falsa.

```
$ cp <web_server_fake_pem.key> /etc/pki/nginx/private/server.key
```

5. Execute o comando a seguir para alterar a propriedade dos arquivos, para que o usuário chamado nginx possa lê-los.

```
$ chown nginx /etc/pki/nginx/server.crt /etc/pki/nginx/private/server.key
```

6. Configure o OpenSSL para usar o provedor. AWS CloudHSM Para obter mais informações sobre como configurar o OpenSSL Provider, consulte [AWS CloudHSM OpenSSL Provider for Client SDK 5](#).

1. Localize seu arquivo de configuração do OpenSSL:

```
$ openssl version -d
```

Você deve ver uma saída semelhante a:

```
OPENSSLDIR: "/etc/pki/tls"
```

O arquivo de configuração está `openssl.cnf` nesse diretório.

- 2.

 Note

Não modifique diretamente o arquivo `openssl.cnf` padrão do seu sistema. Isso evita que as operações OpenSSL em todo o sistema (conexões SSH, TLS e outros serviços) sejam roteadas acidentalmente pelo provedor CloudHSM.

O uso de um arquivo de configuração separado permite que você defina o escopo do uso do CloudHSM Provider somente para aplicativos específicos que exigem operações criptográficas baseadas em HSM.

Crie um novo arquivo de configuração do OpenSSL com o seguinte conteúdo:

```
$ cat > <example-cloudhsm-openssl.cnf> << 'EOF'
## NOTE: This should point to the system default openssl config file.
# Replace /etc/pki/tls with the path to your OpenSSL configuration directory
.include </etc/pki/tls>/openssl.cnf

# Override the existing provider_section to include AWS CloudHSM OpenSSL
  Provider as a 3rd party OpenSSL provider
[provider_sect]
default = default_sect
# Include AWS CloudHSM CloudHSM OpenSSL provider
cloudhsm = cloudhsm_sect

[default_sect]
activate = 1

[cloudhsm_sect]
activate = 1
EOF
```

3. Certifique-se de que a variável de CLOUDHSM_PIN ambiente esteja definida com suas credenciais de usuário criptográfico (CU):

```
$ export CLOUDHSM_PIN=<username>:<password>
```

4. Defina a variável de OPENSSL_CONF ambiente para apontar para seu arquivo de configuração atualizado e verificar se o provedor está carregado:

```
$ OPENSSL_CONF=/path/to/example-cloudhsm-openssl.cnf openssl list -providers
```

Você deve ver o provedor padrão e o provedor do CloudHSM listados:

```
OPENSSL_CONF=/path/to/example-cloudhsm-openssl.cnf openssl list -providers
Providers:
  default
```

```

name: OpenSSL Default Provider
version: 3.2.2
status: active
cloudhsm
name: AWS CloudHSM OpenSSL Provider
version: 5.17.0
status: active

```

7. Execute o comando a seguir para fazer backup do arquivo `/etc/nginx/nginx.conf`.

```
$ cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.backup
```

8. Atualizar a configuração para NGINX.

Note

Cada cluster pode suportar no máximo 1000 processos de trabalho do NGINX em todos os servidores Web do NGINX.

Amazon Linux 2023

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso pode exigir permissões de root do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```

env CLOUDHSM_PIN;
env OPENSSL_CONF;

```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```

# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
    ciphers

```

```
# Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
# ssl_dhparam "/etc/pki/nginx/dhparams.pem";
ssl_session_cache shared:SSL:1m;
ssl_session_timeout 10m;
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
ssl_prefer_server_ciphers off;

# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {
}

error_page 404 /404.html;
location = /40x.html {
}

error_page 500 502 503 504 /50x.html;
location = /50x.html {
}
}
```

RHEL 9 (9.2+)

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso pode exigir permissões de root do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
```



```

listen      443 ssl http2 default_server;
listen      [::]:443 ssl http2 default_server;
server_name _;
root        /usr/share/nginx/html;

ssl_certificate "/etc/pki/nginx/server.crt";
ssl_certificate_key "/etc/pki/nginx/private/server.key";
# It is strongly recommended to generate unique DH parameters for DHE
ciphers
# Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
# ssl_dhparam "/etc/pki/nginx/dhparams.pem";
ssl_session_cache shared:SSL:1m;
ssl_session_timeout 10m;
ssl_protocols TLSv1.2 TLSv1.3;
ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
ssl_prefer_server_ciphers off;

# Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {
}

error_page 404 /404.html;
location = /40x.html {
}

error_page 500 502 503 504 /50x.html;
location = /50x.html {
}
}

```

RHEL 10 (10.0+)

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso pode exigir permissões de root do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
    ciphers
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    # ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
    ssl_prefer_server_ciphers off;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
```

```
}
}
```

Ubuntu 24.04

Use um editor de texto para editar o arquivo `/etc/nginx/nginx.conf`. Isso pode exigir permissões de root do Linux. Na parte superior do arquivo, adicione as seguintes linhas:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Em seguida, adicione o seguinte à seção TLS do arquivo:

```
# Settings for a TLS enabled server.
server {
    listen      443 ssl http2 default_server;
    listen      [::]:443 ssl http2 default_server;
    server_name _;
    root        /var/www/html;

    ssl_certificate "/etc/ssl/certs/server.crt";
    ssl_certificate_key "/etc/ssl/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
    ciphers
    # Generate them with: openssl dhparam -out /etc/ssl/certs/dhparams.pem 2048
    # ssl_dhparam "/etc/ssl/certs/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout 10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
    ssl_prefer_server_ciphers off;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;
```

```
location / {  
}  
  
error_page 404 /404.html;  
location = /40x.html {  
}  
  
error_page 500 502 503 504 /50x.html;  
location = /50x.html {  
}  
}
```

Salve o arquivo.

9. Faça o backup do arquivo de configuração `systemd` e defina o caminho `EnvironmentFile`.

Amazon Linux 2023

1. Faça backup do `nginx.service` arquivo:

```
$ cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra `/lib/systemd/system/nginx.service` em um editor de texto. Na seção [Serviço], adicione:

```
EnvironmentFile=/etc/sysconfig/nginx
```

RHEL 9 (9.2+)

1. Faça backup do `nginx.service` arquivo:

```
$ cp /lib/systemd/system/nginx.service /lib/systemd/system/  
nginx.service.backup
```

2. Abra `/lib/systemd/system/nginx.service` em um editor de texto. Na seção [Serviço], adicione:

```
EnvironmentFile=/etc/sysconfig/nginx
```

RHEL 10 (10.0+)

1. Faça backup do `nginx.service` arquivo:

```
$ cp /lib/systemd/system/nginx.service /lib/systemd/system/nginx.service.backup
```

2. Abra `/lib/systemd/system/nginx.service` em um editor de texto. Na seção [Serviço], adicione:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 24.04

1. Faça backup do `nginx.service` arquivo:

```
$ cp /lib/systemd/system/nginx.service /lib/systemd/system/nginx.service.backup
```

2. Abra `/lib/systemd/system/nginx.service` em um editor de texto. Na seção [Serviço], adicione:

```
EnvironmentFile=/etc/sysconfig/nginx
```

10. Verifique se o arquivo `/etc/sysconfig/nginx` existe e siga um destes procedimentos:

- Se o arquivo existir, faça backup do arquivo executando o seguinte comando:

```
$ cp /etc/sysconfig/nginx /etc/sysconfig/nginx.backup
```

- Se o arquivo não existir, abra um editor de texto e crie um arquivo chamado `nginx` na pasta `/etc/sysconfig/`.

11. Configure o ambiente NGINX.

Amazon Linux 2023

Como usuário-raiz do Linux, abra o arquivo `/etc/sysconfig/nginx` em um editor de texto. Por exemplo,

```
vi /etc/sysconfig/nginx
```

Adicione as credenciais do Cryptography User (CU) e o caminho para seu arquivo de configuração do OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>  
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC. *<path to example-cloudhsm-openssl.cnf>* Substitua pelo caminho completo para o arquivo de configuração que você criou [Para configurar o NGINX para o OpenSSL Provider](#).

Salve o arquivo.

RHEL 9 (9.2+)

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso pode exigir permissões de root do Linux. Adicione as credenciais do Cryptography User (CU) e o caminho para seu arquivo de configuração do OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>  
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC. *<path to example-cloudhsm-openssl.cnf>* Substitua pelo caminho completo para o arquivo de configuração que você criou [Para configurar o NGINX para o OpenSSL Provider](#).

Salve o arquivo.

RHEL 10 (10.0+)

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso pode exigir permissões de root do Linux. Adicione as credenciais do Cryptography User (CU) e o caminho para seu arquivo de configuração do OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>  
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC. *<path to example-cloudhsm-openssl.cnf>* Substitua pelo caminho completo para o arquivo de configuração que você criou [Para configurar o NGINX para o OpenSSL Provider](#).

Salve o arquivo.

Ubuntu 24.04

Abra o arquivo `/etc/sysconfig/nginx` em um editor de textos. Isso pode exigir permissões de root do Linux. Adicione as credenciais do Cryptography User (CU) e o caminho para seu arquivo de configuração do OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Substitua *<CU user name>* e *<password>* pelas credenciais da UC. *<path to example-cloudhsm-openssl.cnf>* Substitua pelo caminho completo para o arquivo de configuração que você criou [Para configurar o NGINX para o OpenSSL Provider](#).

Salve o arquivo.

12. Inicie o servidor web NGINX.

Amazon Linux 2023

Parar todos os processos do NGINX

```
$ systemctl stop nginx
```

Recarregue a configuração `systemd` para receber as alterações mais recentes

```
$ systemctl daemon-reload
```

Iniciar o NGINX

```
$ systemctl start nginx
```

RHEL 9 (9.2+)

Pare qualquer processo NGINX em execução

```
$ systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ systemctl daemon-reload
```

Inicie o processo NGINX

```
$ systemctl start nginx
```

RHEL 10 (10.0+)

Pare qualquer processo NGINX em execução

```
$ systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ systemctl daemon-reload
```

Inicie o processo NGINX

```
$ systemctl start nginx
```

Ubuntu 24.04

Pare qualquer processo NGINX em execução

```
$ systemctl stop nginx
```

Recarregue a configuração systemd para receber as alterações mais recentes

```
$ systemctl daemon-reload
```

Inicie o processo NGINX


```
$ systemctl start nginx
```

Depois de configurar o NGINX, acesse. [Verificar se o HTTPS usa o certificado que você configurou](#)

Configurar servidor HAProxy web

Use esta seção para configurar HAProxy com o OpenSSL Provider. Os exemplos a seguir mostram como configurar seus certificados e HAProxy chaves do CloudHSM.

Para configurar HAProxy para o OpenSSL Provider

1. Faça backup do arquivo de certificado combinado existente, se ele existir:

```
$ cp server-combined.pem server-combined.pem.backup
```

2. Crie um arquivo de certificado combinado para HAProxy usar seu certificado e a chave PEM falsa do CloudHSM:

```
$ cat server.crt server.key > server-combined.pem
```

3. Faça backup da HAProxy configuração existente:

```
$ cp /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg.backup
```

4. Crie uma nova configuração de descarregamento TLS do CloudHSM em: `/etc/haproxy/haproxy.cfg`

```
global
    daemon
    ssl-provider cloudhsm
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/haproxy/dhparams.pem 2048
    # ssl-dh-param-file /etc/haproxy/dhparams.pem
    ssl-default-bind-ciphers ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-
GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-
AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-
SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES256-
GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-CHACHA20-POLY1305
```

```
ssl-default-bind-ciphersuites
TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    mode http
    timeout connect 5000ms
    timeout client 50000ms
    timeout server 50000ms

frontend haproxy_frontend
    bind *:443 ssl crt /path/to/server-combined.pem
    default_backend web_servers

backend web_servers
    server web1 127.0.0.1:8080 check
```

Atualize o caminho do certificado para corresponder ao local do seu arquivo.

5. Configure o systemd para usar um arquivo de ambiente para HAProxy. A localização depende da sua distribuição Linux.

Amazon Linux and RHEL

Faça backup e modifique o arquivo HAProxy de serviço:

```
$ cp /lib/systemd/system/haproxy.service /lib/systemd/system/
haproxy.service.backup
```

Edite `/lib/systemd/system/haproxy.service` e adicione a seguinte linha na seção [Serviço]:

```
EnvironmentFile=/etc/sysconfig/haproxy
```

Ubuntu

Faça backup e modifique o arquivo HAProxy de serviço:

```
$ cp /lib/systemd/system/haproxy.service /lib/systemd/system/
haproxy.service.backup
```

Edite `/lib/systemd/system/haproxy.service` e adicione a seguinte linha na seção [Serviço]:

```
EnvironmentFile=/etc/default/haproxy
```

6. Crie o arquivo de ambiente no local apropriado para seu sistema.

Amazon Linux and RHEL

Faça backup do arquivo do HAProxy ambiente, se ele existir:

```
$ cp /etc/sysconfig/haproxy /etc/sysconfig/haproxy.backup
```

Crie o arquivo de HAProxy ambiente `/etc/sysconfig/haproxy` com o seguinte conteúdo:

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Ubuntu

Faça backup do arquivo do HAProxy ambiente, se ele existir:

```
$ cp /etc/default/haproxy /etc/default/haproxy.backup
```

Crie o arquivo de HAProxy ambiente `/etc/default/haproxy` com o seguinte conteúdo:

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua `<CU user name>` e `<password>` por suas credenciais de UC.

7. Recarregue a configuração do systemd:

```
$ systemctl daemon-reload
```

8. Comece HAProxy com a configuração de descarregamento TLS do CloudHSM:

```
$ systemctl start haproxy
```

Você também pode executar HAProxy diretamente com um arquivo de configuração personalizado:

```
$ haproxy -f /path/to/haproxy-cloudhsm.cfg
```

Depois de configurar HAProxy, vá para [Verificar se o HTTPS usa o certificado que você configurou](#).

Etapa 4: permitir tráfego HTTPS e verificar o certificado

Depois de configurar seu servidor web para SSL/TLS descarga AWS CloudHSM, adicione sua instância de servidor web a um grupo de segurança que permite tráfego HTTPS de entrada. Isso permite que clientes, como navegadores da Web, estabeleçam uma conexão HTTPS com seu servidor Web. Em seguida, faça uma conexão HTTPS com seu servidor web e verifique se ele está usando o certificado que você configurou para SSL/TLS descarga. AWS CloudHSM

Tópicos

- [Habilitar conexões HTTPS de entrada](#)
- [Verificar se o HTTPS usa o certificado que você configurou](#)

Habilitar conexões HTTPS de entrada

Para se conectar ao seu servidor Web a partir de um cliente (como um navegador da Web), crie um grupo de segurança que permita conexões HTTPS de entrada. Especificamente, ele deve permitir conexões TCP de entrada na porta 443. Atribua esse grupo de segurança ao seu servidor Web.

Para criar um grupo de segurança para HTTPS e atribuí-lo ao seu servidor Web

1. Abra o console do Amazon EC2 em <https://console.aws.amazon.com/ec2/>.
2. No painel de navegação, escolha Grupos de segurança.
3. Escolha Create security group (Criar grupo de segurança).
4. Para Create Security Group, faça o seguinte:
 - a. Para Security group name, digite um nome para security group que você está criando.
 - b. (Opcional) Digite uma descrição do security group que você está criando.
 - c. Para VPC, escolha a VPC que contém a instância do Amazon EC2 do servidor Web.
 - d. Selecione Adicionar regra.

- e. Em Tipo, selecione HTTPS na janela suspensa.
 - f. Em Fonte, insira um local de origem.
 - g. Escolha Create security group (Criar grupo de segurança).
5. No painel de navegação, escolha Instâncias.
 6. Marque a caixa de seleção ao lado da sua instância do servidor Web.
 7. Selecione o menu suspenso Ações, na parte superior da página. Selecione Segurança e, em seguida, Alterar grupos de segurança.
 8. Em Grupos de segurança associados, escolha a caixa de pesquisa e escolha o grupo de segurança que você criou para HTTPS. Em seguida, escolha Adicionar grupos de segurança.
 9. Selecione Save (Salvar).

Verificar se o HTTPS usa o certificado que você configurou

Depois de adicionar o servidor web a um grupo de segurança, você pode verificar se o SSL/TLS offload está usando seu certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como [OpenSSL s_client](#).

Para verificar o SSL/TLS descarregamento com um navegador da web

1. Use um navegador da Web para se conectar ao servidor Web usando o nome de DNS público ou endereço IP do servidor. Certifique-se de que a URL na barra de endereços comece com https://. Por exemplo, **`https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`**

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, `https://www.example.com/`) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Use seu navegador da Web para ver o certificado do servidor Web. Para obter mais informações, consulte:
 - Para o Mozilla Firefox, consulte o tópico sobre como [Visualizar um certificado](#), no site de suporte da Mozilla.

- Para o Google Chrome, consulte [Noções básicas de problemas de segurança](#), no site Google Tools for Web Developers.

Outros navegadores da Web podem ter recursos semelhantes, que você pode usar para visualizar o certificado do servidor Web.

3. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Para verificar o SSL/TLS descarregamento com o OpenSSL s_client

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS.
<server name> Substitua pelo nome DNS público ou endereço IP do seu servidor web.

```
openssl s_client -connect <server name>:443
```

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Agora você tem um site protegido com HTTPS. A chave privada do servidor web é armazenada em um HSM no seu AWS CloudHSM cluster.

Para adicionar um balanceador de carga, consulte [Adicione um balanceador de carga com o Elastic Load Balancing AWS CloudHSM para \(opcional\)](#).

AWS CloudHSM Descarregamento de SSL/TLS no Linux usando Tomcat com JSSE

Este tópico fornece step-by-step instruções para configurar o SSL/TLS descarregamento usando o Java Secure Socket Extension (JSSE) com o JCE SDK. AWS CloudHSM

Tópicos

- [Visão geral do](#)
- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: gerar ou importar uma chave privada e um SSL/TLS certificado](#)
- [Etapa 3: configure o servidor Web Tomcat](#)
- [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#)

Visão geral do

Em AWS CloudHSM, os servidores web Tomcat funcionam no Linux para oferecer suporte a HTTPS. O AWS CloudHSM JCE SDK fornece uma interface que pode ser usada com o JSSE (Java Secure Socket Extension) para permitir o uso desses servidores web HSMs. AWS CloudHSM O JCE é a ponte que conecta o JSSE ao seu cluster AWS CloudHSM. O JSSE é uma API Java para os protocolos Transport Layer Security (TLS) e Secure Sockets Layer (SSL).

Etapa 1: configurar os pré-requisitos

Siga estes pré-requisitos para usar um servidor web Tomcat AWS CloudHSM para SSL/TLS descarregamento no Linux. Esses pré-requisitos devem ser atendidos para configurar o SSL/TLS descarregamento do servidor web com o Client SDK 5 e um servidor web Tomcat.

Note

Plataformas diferentes exigem pré-requisitos diferentes. Sempre siga as etapas de instalação corretas para sua plataforma.

Pré-requisitos

- Uma instância do Amazon EC2 executando um sistema operacional Linux com um servidor web Tomcat instalado.
- Um [usuário de criptografia](#) (CU) para ter e gerenciar a chave privada do servidor web no HSM.
- Um AWS CloudHSM cluster ativo com pelo menos dois módulos de segurança de hardware (HSMs) que têm o [JCE for Client SDK 5](#) instalado e configurado.

 Note

Você pode usar um único cluster HSM, mas primeiro deve desativar a durabilidade da chave do cliente. Para obter mais informações, consulte [Gerenciar configurações de durabilidade da chave do cliente](#) e [Ferramenta de configuração do Client SDK 5](#).

Como atender aos pré-requisitos

1. Instale e configure o JCE AWS CloudHSM em um AWS CloudHSM cluster ativo com pelo menos dois módulos de segurança de hardware (HSMs). Para obter mais informações sobre instalação, consulte [JCE para o Client SDK 5](#).
2. Em uma instância do EC2 Linux que tenha acesso ao seu AWS CloudHSM cluster, siga as [instruções do Apache Tomcat](#) para baixar e instalar o servidor web Tomcat.
3. Use a CLI do [CloudHSM](#) para criar um usuário de criptografia (CU). Para obter mais informações sobre o gerenciamento de usuários do HSM, consulte [Gerenciar usuários do HSM com a CLI do CloudHSM](#).

 Tip

Lembre o nome do usuário e a senha do CU. Eles serão necessários mais tarde ao gerar ou importar a chave privada HTTPS e o certificado para o servidor web.

4. Para configurar o JCE com o Java Keytool, siga as instruções em [Use o Client SDK 5 para fazer a integração AWS CloudHSM com o Java Keytool e o Jarsigner](#).

Depois de concluir essas etapas, vá para [Etapa 2: gerar ou importar uma chave privada e um SSL/TLS certificado](#).

Observações

- Para usar o Security-Enhanced Linux (SELinux) e servidores web, você deve permitir conexões TCP de saída na porta 2223, que é a porta que o Client SDK 5 usa para se comunicar com o HSM.
- Para criar e ativar um cluster e dar acesso a uma instância do EC2 ao cluster, conclua as etapas em [Conceitos básicos de AWS CloudHSM](#). Esta seção oferece step-by-step instruções para criar

um cluster ativo com um HSM e uma instância cliente do Amazon EC2. Você pode usar essa instância de cliente como seu servidor Web.

- Para evitar a desativação da durabilidade da chave do cliente, adicione mais de um HSM ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
- Você pode usar um cliente SSH ou PuTTY para se conectar à instância do cliente. Para obter mais informações, consulte [Conectar à instância Linux utilizando SSH](#) ou [Conectar à instância Linux no Windows utilizando PuTTY](#) na documentação do Amazon C2.

Etapa 2: gerar ou importar uma chave privada e um SSL/TLS certificado

Para habilitar o HTTPS, seu aplicativo de servidor web Tomcat precisa de uma chave privada e um SSL/TLS certificado correspondente. Para usar o servidor web SSL/TLS offload with AWS CloudHSM, você deve armazenar a chave privada em um HSM no seu cluster. AWS CloudHSM

Note

Se você ainda não tem uma chave privada e um certificado correspondente, gere uma chave privada em um HSM. Você usa a chave privada para criar uma solicitação de assinatura de certificado (CSR), que você usa para criar o SSL/TLS certificado.

Você cria um AWS CloudHSM KeyStore arquivo local que contém uma referência à sua chave privada no HSM e no certificado associado. Seu servidor web usa o AWS CloudHSM KeyStore arquivo para identificar a chave privada no HSM durante o SSL/TLS descarregamento.

Tópicos

- [Gerar uma chave privada](#)
- [Gere um certificado autoassinado](#)

Gerar uma chave privada

Esta seção mostra como gerar um par de chaves usando o KeyTool do JDK. Depois de gerar um par de chaves dentro do HSM, você pode exportá-lo como um KeyStore arquivo e gerar o certificado correspondente.

Dependendo do seu caso de uso, você pode gerar um RSA ou um par de chaves EC. As etapas a seguir mostram como gerar um par de chaves RSA.

Use o **genkeypair** comando in KeyTool para gerar um par de chaves RSA

1. Depois de substituir os dados **<VARIABLES>** abaixo por seus dados específicos, use o comando a seguir para gerar um arquivo de armazenamento de chaves chamado `jsse_keystore.keystore`, que terá uma referência de sua chave privada no HSM.

```
$ keytool -genkeypair -alias <UNIQUE ALIAS FOR KEYS> -keyalg <KEY ALGORITHM> -
keysize <KEY SIZE> -sigalg <SIGN ALGORITHM> \
    -keystore <PATH>/<JSSE KEYSTORE NAME>.keystore -storetype CLOUDHSM \
    -dname CERT_DOMAIN_NAME \
    -J-classpath '-J'$JAVA_LIB'/*:/opt/cloudhsm/java/*:./*' \
    -provider "com.amazonaws.cloudhsm.jce.provider.CloudHsmProvider" \
    -providerpath "$CLOUDHSM_JCE_LOCATION" \
    -keypass <KEY PASSWORD> -storepass <KEYSTORE PASSWORD>
```

- **<PATH>**: o caminho pelo qual você deseja gerar seu arquivo de armazenamento de chaves.
 - **<UNIQUE ALIAS FOR KEYS>**: Isso é usado para identificar de forma exclusiva sua chave no HSM. Esse alias será definido como o atributo RÓTULO da chave.
 - **<KEY PASSWORD>**: armazenamos a referência à sua chave no arquivo de armazenamento de chaves local, e essa senha protege essa referência local.
 - **<KEYSTORE PASSWORD>**: essa é a senha do seu arquivo de armazenamento de chaves local.
 - **<JSSE KEYSTORE NAME>**: nome do arquivo Keystore.
 - **<CERT DOMAIN NAME>**: X.500 Nome distinto.
 - **<KEY ALGORITHM>**: Algoritmo de chave para gerar o par de chaves (por exemplo, RSA e EC).
 - **<KEY SIZE>**: tamanho da chave para gerar o par de chaves (por exemplo, 2048, 3072 e 4096).
 - **<SIGN ALGORITHM>**: Tamanho da chave para gerar o par de chaves (por exemplo, SHA1with RSA, SHA224with RSA, SHA256with RSA e SHA384with RSA). SHA512with
2. Para confirmar se o comando foi bem-sucedido, digite o comando a seguir e verifique se você gerou com êxito um par de chaves RSA.

```
$ ls <PATH>/<JSSE KEYSTORE NAME>.keystore
```

Gere um certificado autoassinado

Depois de gerar uma chave privada junto com o arquivo de armazenamento de chaves, você pode usar esse arquivo para gerar uma solicitação de assinatura de certificado (CSR) e um certificado.

Em um ambiente de produção, geralmente usa-se uma autoridade de certificação (CA) para criar um certificado de uma CSR. Não é necessária uma CA para um ambiente de teste. Se você usa uma CA, envie o arquivo CSR para eles e use o SSL/TLS certificado assinado que eles fornecem em seu servidor web para HTTPS.

Como alternativa ao uso de uma CA, você pode usar o KeyTool para criar um certificado autoassinado. Os certificados autoassinados não são confiáveis para os navegadores e não devem ser usados em ambientes de produção. Eles podem ser usados em ambientes de teste.

Warning

Os certificados autoassinados devem ser usados apenas em um ambiente de teste. Para um ambiente de produção, use um método mais seguro, como uma autoridade de certificação, para criar um certificado.

Gere um certificado

1. Obtenha uma cópia do arquivo de armazenamento de chaves gerado em uma etapa anterior.
2. Execute o comando a seguir para usar o KeyTool para criar uma solicitação de assinatura de certificado (CSR).

```
$ keytool -certreq -keyalg RSA -alias unique_alias_for_key -file certreq.csr \  
-keystore <JSSE KEYSTORE NAME>.keystore -storetype CLOUDHSM \  
-J-classpath '-J$JAVA_LIB/*:/opt/cloudhsm/java/*:./*' \  
-keypass <KEY PASSWORD> -storepass <KEYSTORE PASSWORD>
```

Note

O arquivo de saída da solicitação de assinatura do certificado é `certreq.csr`.

Assine um certificado

- Depois de substituir os dados **<VARIABLES>** abaixo por seus dados específicos, execute o comando a seguir para assinar sua CSR com sua chave privada em seu HSM. Isso cria um certificado autoassinado.

```
$ keytool -gencert -infile certreq.csr -outfile certificate.crt \
  -alias <UNIQUE ALIAS FOR KEYS> -keypass <KEY_PASSWORD> -
storepass <KEYSTORE_PASSWORD> -sigalg SIG_ALG \
  -storetype CLOUDHSM -J-classpath '-J$JAVA_LIB/*:/opt/cloudhsm/java/*:./*' \
  -keystore jsse_keystore.keystore
```

Note

certificate.crt é o certificado assinado que usa a chave privada do alias.

Importar um certificado no Keystore

- Depois de substituir os dados **<VARIABLES>** abaixo pelos seus dados específicos, execute o comando a seguir para importar um certificado assinado como um certificado confiável. Esta etapa armazenará o certificado na entrada do keystore identificada pelo alias.

```
$ keytool -import -alias <UNIQUE ALIAS FOR KEYS> -keystore jsse_keystore.keystore \
  -file certificate.crt -storetype CLOUDHSM \
  -v -J-classpath '-J$JAVA_LIB/*:/opt/cloudhsm/java/*:./*' \
  -keypass <KEY_PASSWORD> -storepass <KEYSTORE_PASSWORD>
```

Converta um certificado em um PEM

- Execute o comando a seguir para converter o arquivo de certificado assinado (.crt) em um PEM. O arquivo PEM será usado para enviar a solicitação do cliente http.

```
$ openssl x509 -inform der -in certificate.crt -out certificate.pem
```

Depois de concluir estas etapas, vá para a [Etapa 3: Configure o servidor web](#).

Etapa 3: configure o servidor Web Tomcat

Atualize a configuração do software de servidor web para usar o certificado HTTPS e a chave privada PEM falsa correspondente que você criou na etapa anterior. Lembre-se de fazer backup de seus certificados e chaves existentes antes de começar. Isso concluirá a configuração do software do servidor web Linux para SSL/TLS descarta com. AWS CloudHSM Para obter mais informações, consulte a [Referência de configuração do Apache Tomcat 9](#).

Interrompa o servidor

- Depois de substituir os dados **<VARIABLES>** abaixo pelos seus dados específicos, execute o seguinte comando para parar o servidor Tomcat antes de atualizar a configuração

```
$ /<TOMCAT DIRECTORY>/bin/shutdown.sh
```

- <TOMCAT DIRECTORY>**: Seu diretório de instalação do Tomcat.

Atualizar o classpath do Tomcat

- Conecte-se à instância do cliente.
- Localize a pasta de instalação do Tomcat.
- Depois de substituir os dados **<VARIABLES>** abaixo por seus dados específicos, use o comando a seguir para adicionar a biblioteca Java e o caminho AWS CloudHSM Java no Tomcatclasspath, localizados no Tomcat/bin/catalina arquivo.sh.

```
$ sed -i 's@CLASSPATH="$CLASSPATH"$CATALINA_HOME"/bin/bootstrap.jar@CLASSPATH="$CLASSPATH"$CATALINA_HOME"/bin/bootstrap.jar:"<JAVA LIBRARY>"\/*:/opt/cloudhsm/java/*:.//*' <TOMCAT PATH> /bin/catalina.sh
```

- <JAVA LIBRARY>**: localização da biblioteca Java JRE.
- <TOMCAT PATH>**: pasta de instalação do Tomcat.

Adicione um conector HTTPS na configuração do servidor.

- Vá até a pasta de instalação do Tomcat.

- Depois de substituir os dados **<VARIABLES>** abaixo pelos seus dados específicos, use o comando a seguir para adicionar um conector HTTPS para usar certificados gerados nos pré-requisitos:

```
$ sed -i '/<Connector port="8080"/i <Connector port="\443\" maxThreads="\200\"
scheme="\https\" secure="\true\" SSLEnabled="\true\" keystoreType="\CLOUDHSM\"
keystoreFile="\
    <CUSTOM DIRECTORY>/<JSSE KEYSTORE NAME>.keystore\" keystorePass="\<KEYSTORE
PASSWORD>\" keyPass="\<KEY PASSWORD>
    \" keyAlias="\<UNIQUE ALIAS FOR KEYS>" clientAuth="\false\" sslProtocol=
"\TLS\"/>' <TOMCAT PATH>/conf/server.xml
```

- <CUSTOM DIRECTORY>**: Diretório em que o arquivo keystore está localizado.
- <JSSE KEYSTORE NAME>**: nome do arquivo Keystore.
- <KEYSTORE PASSWORD>**: essa é a senha do seu arquivo de armazenamento de chaves local.
- <KEY PASSWORD>**: armazenamos a referência à sua chave no arquivo de armazenamento de chaves local, e essa senha protege essa referência local.
- <UNIQUE ALIAS FOR KEYS>**: Isso é usado para identificar de forma exclusiva sua chave no HSM. Esse alias será definido como o atributo RÓTULO da chave.
- <TOMCAT PATH>**: O caminho para sua pasta Tomcat.

Iniciar o servidor

- Depois de substituir os dados **<VARIABLES>** abaixo pelos seus dados específicos, use o comando a seguir para iniciar o servidor Tomcat:

```
$ ./<TOMCAT DIRECTORY>/bin/startup.sh
```

Note

<TOMCAT DIRECTORY> é o nome do seu diretório de instalação do Tomcat.

Depois de atualizar a configuração do servidor web, vá para [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#).

Etapa 4: permitir tráfego HTTPS e verificar o certificado

Depois de configurar seu servidor web para SSL/TLS descarga AWS CloudHSM, adicione sua instância de servidor web a um grupo de segurança que permite tráfego HTTPS de entrada. Isso permite que clientes, como navegadores da Web, estabeleçam uma conexão HTTPS com seu servidor Web. Em seguida, faça uma conexão HTTPS com seu servidor web e verifique se ele está usando o certificado que você configurou para SSL/TLS descarga. AWS CloudHSM

Tópicos

- [Habilitar conexões HTTPS de entrada](#)
- [Verificar se o HTTPS usa o certificado que você configurou](#)

Habilitar conexões HTTPS de entrada

Para se conectar ao seu servidor Web a partir de um cliente (como um navegador da Web), crie um grupo de segurança que permita conexões HTTPS de entrada. Especificamente, ele deve permitir conexões TCP de entrada na porta 443. Atribua esse grupo de segurança ao seu servidor Web.

Para criar um grupo de segurança para HTTPS e atribuí-lo ao seu servidor Web

1. Abra o console do Amazon EC2 em <https://console.aws.amazon.com/ec2/>.
2. No painel de navegação, escolha Grupos de segurança.
3. Escolha Create security group (Criar grupo de segurança).
4. Para Create Security Group, faça o seguinte:
 - a. Para Security group name, digite um nome para security group que você está criando.
 - b. (Opcional) Digite uma descrição do security group que você está criando.
 - c. Para VPC, escolha a VPC que contém a instância do Amazon EC2 do servidor Web.
 - d. Selecione Adicionar regra.
 - e. Em Tipo, selecione HTTPS na janela suspensa.
 - f. Em Fonte, insira um local de origem.
 - g. Escolha Create security group (Criar grupo de segurança).
5. No painel de navegação, escolha Instâncias.
6. Marque a caixa de seleção ao lado da sua instância do servidor Web.

7. Selecione o menu suspenso Ações, na parte superior da página. Selecione Segurança e, em seguida, Alterar grupos de segurança.
8. Em Grupos de segurança associados, escolha a caixa de pesquisa e escolha o grupo de segurança que você criou para HTTPS. Em seguida, escolha Adicionar grupos de segurança.
9. Selecione Save (Salvar).

Verificar se o HTTPS usa o certificado que você configurou

Depois de adicionar o servidor web a um grupo de segurança, você pode verificar se o SSL/TLS offload está usando seu certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como [OpenSSL s_client](#).

Para verificar o SSL/TLS descarregamento com um navegador da web

1. Use um navegador da Web para se conectar ao servidor Web usando o nome de DNS público ou endereço IP do servidor. Certifique-se de que a URL na barra de endereços comece com https://. Por exemplo, **https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/**

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Use seu navegador da Web para ver o certificado do servidor Web. Para obter mais informações, consulte:
 - Para o Mozilla Firefox, consulte o tópico sobre como [Visualizar um certificado](#), no site de suporte da Mozilla.
 - Para o Google Chrome, consulte [Noções básicas de problemas de segurança](#), no site Google Tools for Web Developers.

Outros navegadores da Web podem ter recursos semelhantes, que você pode usar para visualizar o certificado do servidor Web.

3. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Para verificar o SSL/TLS descarregamento com o OpenSSL s_client

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS.
<server name> Substitua pelo nome DNS público ou endereço IP do seu servidor web.

```
openssl s_client -connect <server name>:443
```

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Agora você tem um site protegido com HTTPS. A chave privada do servidor web é armazenada em um HSM no seu AWS CloudHSM cluster.

Para adicionar um balanceador de carga, consulte [Adicione um balanceador de carga com o Elastic Load Balancing AWS CloudHSM para \(opcional\)](#).

AWS CloudHSM Descarregamento de SSL/TLS no Windows usando IIS com KSP

Este tutorial fornece step-by-step instruções para configurar o SSL/TLS offload AWS CloudHSM em um servidor web Windows.

Tópicos

- [Visão geral do](#)
- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: Criar uma solicitação de assinatura de certificado \(CSR\) e um certificado](#)

- [Etapa 3: configure o servidor Web](#)
- [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#)

Visão geral do

No Windows, o aplicativo do servidor web [Serviços de Informações da Internet \(IIS\) para Windows Server](#) oferece suporte nativo a HTTPS. O [provedor de armazenamento de AWS CloudHSM chaves \(KSP\) da API de criptografia: Next Generation \(CNG\) da Microsoft](#) fornece a interface que permite que o IIS use os HSMs em seu cluster para descarga criptográfica e armazenamento de chaves. O AWS CloudHSM KSP é a ponte que conecta o IIS ao seu AWS CloudHSM cluster.

Este tutorial mostra como fazer o seguinte:

- Instale o software do servidor web em uma instância do Amazon EC2.
- Configure o software do servidor Web para oferecer suporte a HTTPS com uma chave privada armazenada em seu AWS CloudHSM cluster.
- (Opcional) Use o Amazon EC2 para criar uma segunda instância de servidor Web e o Elastic Load Balancing para criar um balanceador de carga. Usar um load balanceador de carga pode aumentar o desempenho, distribuindo a carga em vários servidores. Ele também pode fornecer redundância e maior disponibilidade se um ou mais servidores falhar.

Quando estiver pronto para começar, vá para [Etapa 1: configurar os pré-requisitos](#).

Etapa 1: configurar os pré-requisitos

Plataformas diferentes exigem pré-requisitos diferentes. Use a seção de pré-requisitos abaixo que corresponde à sua plataforma.

Tópicos

- [Pré-requisitos para o Client SDK 5](#)
- [Pré-requisitos para o Client SDK 3](#)

Pré-requisitos para o Client SDK 5

Para configurar o servidor web SSL/TLS offload com AWS CloudHSM, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos um HSM.

- Uma instância do Amazon EC2 executando um sistema operacional Windows com o seguinte software instalado:
 - O software AWS CloudHSM cliente para Windows.
 - Serviços de Informações da Internet (IIS) para Windows Server.
- Um [usuário de criptografia](#) (CU) para ter e gerenciar a chave privada do servidor web no HSM.

 Note

Este tutorial usa o Microsoft Windows Server 2019. O Microsoft Windows Server 2016 e o 2022 também são compatíveis.

Para configurar uma instância do Windows Server e criar um CU no HSM

1. Siga as etapas em [Introdução](#). Quando você iniciar o cliente do Amazon EC2, escolha um Windows Server 2019 AMI. Quando você concluir estas etapas, terá um cluster ativo com um HSM, no mínimo. Você também tem uma instância cliente do Amazon EC2 executando o Windows Server com o software AWS CloudHSM cliente para Windows instalado.
2. (Opcional) Adicione mais HSMs ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
3. Conecte-se ao servidor do Windows. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
4. Use a CLI do CloudHSM para criar um usuário de criptografia (CU). Lembre o nome do usuário e a senha do CU. Você precisará deles para concluir a próxima etapa.

 Note

Para obter informações sobre como criar um usuário, consulte [Gerenciamento de usuários do HSM com a CLI do CloudHSM](#).

5. [Defina as credenciais de login para o HSM](#), usando o nome de usuário e a senha do CU que você criou na etapa anterior.
6. Na etapa 5, se você usou o Gerenciador de Credenciais do Windows para definir as credenciais [psexec.exe](#) do HSM, faça o download SysInternals para executar o seguinte comando como NT Authority\SYSTEM:

```
psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools\set_cloudhsm_credentials.exe"  
--username <USERNAME> --password <PASSWORD>
```

Substitua <USERNAME> e <PASSWORD> pelas credenciais do HSM.

Para instalar o IIS no Windows Server

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
2. No Windows Server, inicie o Gerenciador de servidores.
3. No painel Server Manager, escolha Add roles and features.
4. Leia as informações de Before you begin e escolha Next.
5. Em Installation Type (Tipo de instalação), escolha Role-based or feature-based installation (Instalação com base na função ou no recurso). Em seguida, escolha Next (Próximo).
6. Em Server Selection, escolha Select a server from the server pool. Escolha Próximo.
7. Em Server Roles, faça o seguinte:
 - a. Selecione Servidor web (IIS).
 - b. Em Adicionar recursos que são necessários para o Servidor Web (IIS), escolha Add Features (Adicionar recursos).
 - c. Escolha Next para terminar de selecionar funções de servidor.
8. Em Features (Recursos), aceite os padrões. Escolha Próximo.
9. Leia as informações sobre a função Servidor Web (IIS). Escolha Próximo.
10. Em Select role services (Selecionar serviços de função), aceite os valores padrão ou altere as configurações como preferir. Escolha Próximo.
11. Em Confirmation (Confirmação), leia as informações de confirmação. Em seguida, selecione Install (Instalar).
12. Depois que a instalação for concluída, escolha Close (Fechar).

Depois de concluir essas etapas, vá para [Etapa 2: Criar uma solicitação de assinatura de certificado \(CSR\) e um certificado](#).

Pré-requisitos para o Client SDK 3

Para configurar o servidor web SSL/TLS offload com AWS CloudHSM, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos um HSM.
- Uma instância do Amazon EC2 executando um sistema operacional Windows com o seguinte software instalado:
 - O software AWS CloudHSM cliente para Windows.
 - Serviços de Informações da Internet (IIS) para Windows Server.
- Um [usuário de criptografia](#) (CU) para ter e gerenciar a chave privada do servidor web no HSM.

Note

Este tutorial usa o Microsoft Windows Server 2016. O Microsoft Windows Server 2012 também é compatível, mas o Microsoft Windows Server 2012 R2 não é.

Para configurar uma instância do Windows Server e criar um CU no HSM

1. Siga as etapas em [Introdução](#). Quando você ativar o cliente do Amazon EC2, escolha uma AMI do Windows Server 2016 ou Windows Server 2012. Quando você concluir estas etapas, terá um cluster ativo com um HSM, no mínimo. Você também tem uma instância cliente do Amazon EC2 executando o Windows Server com o software AWS CloudHSM cliente para Windows instalado.
2. (Opcional) Adicione mais HSMs ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
3. Conecte-se ao servidor do Windows. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
4. Use a CLI do CloudHSM para criar um usuário de criptografia (CU). Lembre o nome do usuário e a senha do CU. Você precisará deles para concluir a próxima etapa.

Note

Para obter informações sobre como criar um usuário, consulte [Gerenciamento de usuários do HSM com a CLI do CloudHSM](#).

5. [Defina as credenciais de login para o HSM](#), usando o nome de usuário e a senha do CU que você criou na etapa anterior.
6. Na etapa 5, se você usou o Gerenciador de Credenciais do Windows para definir as credenciais [psexec.exe](#) do HSM, faça o download SysInternals para executar o seguinte comando como NT Authority\ SYSTEM:

```
psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools\set_cloudhsm_credentials.exe"  
--username <USERNAME> --password <PASSWORD>
```

Substitua <USERNAME> e <PASSWORD> pelas credenciais do HSM.

Para instalar o IIS no Windows Server

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
2. No Windows Server, inicie o Gerenciador de servidores.
3. No painel Server Manager, escolha Add roles and features.
4. Leia as informações de Before you begin e escolha Next.
5. Em Installation Type (Tipo de instalação), escolha Role-based or feature-based installation (Instalação com base na função ou no recurso). Em seguida, escolha Next (Próximo).
6. Em Server Selection, escolha Select a server from the server pool. Escolha Próximo.
7. Em Server Roles, faça o seguinte:
 - a. Selecione Servidor web (IIS).
 - b. Em Adicionar recursos que são necessários para o Servidor Web (IIS), escolha Add Features (Adicionar recursos).
 - c. Escolha Next para terminar de selecionar funções de servidor.
8. Em Features (Recursos), aceite os padrões. Escolha Próximo.
9. Leia as informações sobre a função Servidor Web (IIS). Escolha Próximo.
10. Em Select role services (Selecionar serviços de função), aceite os valores padrão ou altere as configurações como preferir. Escolha Próximo.
11. Em Confirmation (Confirmação), leia as informações de confirmação. Em seguida, selecione Install (Instalar).

12. Depois que a instalação for concluída, escolha Close (Fechar).

Depois de concluir essas etapas, vá para [Etapa 2: Criar uma solicitação de assinatura de certificado \(CSR\) e um certificado](#).

Etapa 2: Criar uma solicitação de assinatura de certificado (CSR) e um certificado

Para habilitar o HTTPS, seu servidor web precisa de um SSL/TLS certificado e uma chave privada correspondente. Para usar o SSL/TLS offload with AWS CloudHSM, você armazena a chave privada no HSM do seu cluster. AWS CloudHSM Para fazer isso, use o [AWS CloudHSM key storage provider \[provedor de armazenamento de chaves \(KSP\)\] para a Cryptography API: Next Generation \(CNG\) da Microsoft](#) para criar uma solicitação de assinatura de certificado (CSR). Em seguida, você fornece a CSR para uma autoridade de certificação (CA), que assina a CSR para produzir um certificado.

Tópicos

- [Criar uma CSR com Client SDK 5](#)
- [Criar uma CSR com Client SDK 3](#)
- [Obtenção e importação de um certificado assinado](#)

Criar uma CSR com Client SDK 5

1. No Windows Server, use um editor de texto para criar um arquivo de solicitação de certificado chamado `IISCertRequest.inf`. O exemplo a seguir mostra o conteúdo de um arquivo `IISCertRequest.inf` de exemplo. Para obter mais informações sobre as seções, as chaves e os valores que você pode especificar no arquivo, consulte a [documentação da Microsoft](#). Não mude o valor `ProviderName`.

```
[Version]
Signature = "$Windows NT$"
[NewRequest]
Subject = "CN=example.com,C=US,ST=Washington,L=Seattle,O=ExampleOrg,OU=WebServer"
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "CloudHSM Key Storage Provider"
KeyUsage = 0xf0
MachineKeySet = True
[EnhancedKeyUsageExtension]
```

```
OID=1.3.6.1.5.5.7.3.1
```

2. Use o [comando certreq do Windows](#) para criar uma CSR no arquivo `IISCertRequest.inf` que você criou na etapa anterior. O exemplo a seguir salva a CSR em um arquivo chamado `IISCertRequest.csr`. Se você usou um nome de arquivo diferente para seu arquivo de solicitação de certificado, *IISCertRequest.inf* substitua pelo nome de arquivo apropriado. Opcionalmente, você pode *IISCertRequest.csr* substituir o arquivo CSR por um nome de arquivo diferente.

```
C:\>certreq -new IISCertRequest.inf IISCertRequest.csr
```

```
CertReq: Request Created
```

O arquivo `IISCertRequest.csr` contém sua CSR. Você precisa dessa CSR para obter um certificado assinado.

Criar uma CSR com Client SDK 3

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
2. Use o comando a seguir para iniciar o AWS CloudHSM daemon do cliente.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```


RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe  
C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

3. No Windows Server, use um editor de texto para criar um arquivo de solicitação de certificado chamado `IISCertRequest.inf`. O exemplo a seguir mostra o conteúdo de um arquivo `IISCertRequest.inf` de exemplo. Para obter mais informações sobre as seções, as chaves e os valores que você pode especificar no arquivo, consulte a [documentação da Microsoft](#). Não mude o valor `ProviderName`.

```
[Version]  
Signature = "$Windows NT$"  
[NewRequest]  
Subject = "CN=example.com,C=US,ST=Washington,L=Seattle,O=ExampleOrg,OU=WebServer"  
HashAlgorithm = SHA256  
KeyAlgorithm = RSA
```

```
KeyLength = 2048
ProviderName = "Cavium Key Storage Provider"
KeyUsage = 0xf0
MachineKeySet = True
[EnhancedKeyUsageExtension]
OID=1.3.6.1.5.5.7.3.1
```

4. Use o [comando certreq do Windows](#) para criar uma CSR no arquivo `IISCertRequest.inf` que você criou na etapa anterior. O exemplo a seguir salva a CSR em um arquivo chamado `IISCertRequest.csr`. Se você usou um nome de arquivo diferente para seu arquivo de solicitação de certificado, *IISCertRequest.inf* substitua pelo nome de arquivo apropriado. Opcionalmente, você pode *IISCertRequest.csr* substituir o arquivo CSR por um nome de arquivo diferente.

```
C:\>certreq -new IISCertRequest.inf IISCertRequest.csr
      SDK Version: 2.03

CertReq: Request Created
```

O arquivo `IISCertRequest.csr` contém sua CSR. Você precisa dessa CSR para obter um certificado assinado.

Obtenção e importação de um certificado assinado

Em um ambiente de produção, geralmente usa-se uma autoridade de certificação (CA) para criar um certificado de uma CSR. Não é necessária uma CA para um ambiente de teste. Se você usa uma CA, envie o arquivo CSR (`IISCertRequest.csr`) para ela e use a CA para criar um SSL/TLS certificado assinado.

Em vez de usar uma CA, você pode usar uma ferramenta como a [OpenSSL](#) para criar um certificado autoassinado.

Warning

Os certificados autoassinados não são confiáveis para os navegadores e não devem ser usados em ambientes de produção. Eles podem ser usados em ambientes de teste.

Os procedimentos a seguir mostram como criar um certificado autoassinado e usá-lo para assinar a CSR do servidor web.

Para criar um certificado autoassinado

1. Use o comando OpenSSL a seguir para criar uma chave privada. Opcionalmente, você pode *SelfSignedCA.key* substituir pelo nome do arquivo para conter sua chave privada.

```
openssl genrsa -aes256 -out SelfSignedCA.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for SelfSignedCA.key:
Verifying - Enter pass phrase for SelfSignedCA.key:
```

2. Use o comando OpenSSL a seguir para criar um certificado autoassinado usando a chave privada que você criou na etapa anterior. Este é um comando interativo. Leia as instruções na tela e siga os avisos. *SelfSignedCA.key* Substitua pelo nome do arquivo que contém sua chave privada (se for diferente). Opcionalmente, você pode *SelfSignedCA.crt* substituir pelo nome do arquivo para conter seu certificado autoassinado.

```
openssl req -new -x509 -days 365 -key SelfSignedCA.key -out SelfSignedCA.crt
Enter pass phrase for SelfSignedCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Para usar o certificado autoassinado para assinar a CSR do servidor web

- Use o comando OpenSSL a seguir para usar a chave privada e o certificado autoassinado para assinar a CSR. Substitua as opções a seguir pelo nome do arquivo que contém os dados correspondentes (se forem diferentes).
 - *IISCertRequest.csr*— O nome do arquivo que contém o CSR do seu servidor web
 - *SelfSignedCA.crt*— O nome do arquivo que contém seu certificado autoassinado
 - *SelfSignedCA.key*— O nome do arquivo que contém sua chave privada
 - *IISCert.crt*— O nome do arquivo que contém o certificado assinado do seu servidor web

```
openssl x509 -req -days 365 -in IISCertRequest.csr \  
              -CA SelfSignedCA.crt \  
              -CAkey SelfSignedCA.key \  
              -CAcreateserial \  
              -out IISCert.crt  
  
Signature ok  
subject=/ST=IIS-HSM/L=IIS-HSM/OU=IIS-HSM/O=IIS-HSM/CN=IIS-HSM/C=IIS-HSM  
Getting CA Private Key  
Enter pass phrase for SelfSignedCA.key:
```

Assim que concluir a etapa anterior, você terá um certificado assinado para seu servidor web (*IISCert.crt*) e um certificado autoassinado (*SelfSignedCA.crt*). Quando tiver esses arquivos, vá para [Etapa 3: configure o servidor Web](#).

Etapa 3: configure o servidor Web

Atualize a configuração do site do IIS para usar o certificado HTTPS que você criou no final da [etapa anterior](#). Isso concluirá a configuração do software de servidor web do Windows (IIS) para SSL/TLS descarta com. AWS CloudHSM

Se tiver usado um certificado autoassinado para assinar a CSR, deverá primeiro importar o certificado autoassinado para Autoridades de certificação raiz confiáveis do Windows.

Para importar o certificado autoassinado para Autoridades de certificação raiz confiáveis do Windows

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.

2. Copie o certificado autoassinado no servidor Windows.
3. No Windows Server, abra o Control Panel (Painel de Controle).
4. Em Search Control Panel (Pesquisar painel de controle), digite **certificates**. Em seguida, escolha Manage computer certificates (Gerenciar certificados de computador).
5. Na janela Certificados – Computador local, clique duas vezes em Autoridades de certificação raiz confiáveis.
6. Clique com o botão direito do mouse em Certificates (Certificados) e escolha Todas as tarefas, Importar.
7. Em Certificate Import Wizard (Assistente para Importação de Certificados), escolha Next (Próximo).
8. Escolha Browse (Procurar) e em seguida localize e selecione o certificado autoassinado. Se tiver criado o certificado autoassinado seguindo as instruções na [etapa anterior deste tutorial](#), o nome do certificado autoassinado será SelfSignedCA.crt. Escolha Open (Abrir).
9. Escolha Next (Próximo).
10. Em Certificate Store (Repositório de Certificados), escolha Colocar todos os certificados no repositório a seguir. Em seguida, confirme se Trusted Root Certification Authorities (Autoridades de certificação raiz confiáveis) está selecionada para Certificate store (Repositório de Certificados).
11. Escolha Next (Próximo) e, em seguida, escolha Finish (Concluir).

Para atualizar a configuração do site do IIS

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
2. Inicie o AWS CloudHSM daemon do cliente.
3. Copie o certificado assinado do servidor web, aquele que você criou no final da [etapa anterior deste tutorial](#) para o servidor Windows.
4. No Windows Server, use o [comando certreq do Windows](#) para aceitar o certificado assinado, como no exemplo a seguir. **IISCert.crt** Substitua pelo nome do arquivo que contém o certificado assinado do seu servidor web.

```
C:\>certreq -accept IISCert.crt  
SDK Version: 2.03
```

5. No Windows Server, inicie o Server Manager (Gerenciador de servidores).

6. No painel Server Manager (Gerenciador de Servidores), no canto superior direito, escolha Tools (Ferramentas), Internet Information Services (IIS) Manager [Gerenciador dos Serviços de Informações da Internet (IIS)].
7. Na janela Internet Information Services (IIS) Manager [Gerenciador dos Serviços de Informações da Internet (IIS)], clique duas vezes no nome do servidor. Em seguida, clique duas vezes em Sites. Selecione o site.
8. Selecione SSL Settings (Configurações SSL). Em seguida, no lado direito da janela, escolha Associações.
9. Na janela Site Bindings (Associações de Site), escolha Add (Adicionar).
10. Em Type (Tipo), escolha https. Em SSL certificate (Certificado SSL), escolha o certificado HTTPS que você criou no final da [etapa anterior deste tutorial](#).

 Note

Se você encontrar um erro durante essa vinculação de certificado, reinicie seu servidor e repita essa etapa.

11. Escolha OK.

Depois que atualizar a configuração do site, vá para [Etapa 4: permitir tráfego HTTPS e verificar o certificado](#).

Etapa 4: permitir tráfego HTTPS e verificar o certificado

Depois de configurar seu servidor web para SSL/TLS descargando AWS CloudHSM, adicione sua instância de servidor web a um grupo de segurança que permite tráfego HTTPS de entrada. Isso permite que clientes, como navegadores da Web, estabeleçam uma conexão HTTPS com seu servidor Web. Em seguida, faça uma conexão HTTPS com seu servidor web e verifique se ele está usando o certificado que você configurou para SSL/TLS descargando AWS CloudHSM.

Tópicos

- [Habilitar conexões HTTPS de entrada](#)
- [Verificar se o HTTPS usa o certificado que você configurou](#)

Habilitar conexões HTTPS de entrada

Para se conectar ao seu servidor Web a partir de um cliente (como um navegador da Web), crie um grupo de segurança que permita conexões HTTPS de entrada. Especificamente, ele deve permitir conexões TCP de entrada na porta 443. Atribua esse grupo de segurança ao seu servidor Web.

Para criar um grupo de segurança para HTTPS e atribuí-lo ao seu servidor Web

1. Abra o console do Amazon EC2 em <https://console.aws.amazon.com/ec2/>.
2. No painel de navegação, escolha Grupos de segurança.
3. Escolha Create security group (Criar grupo de segurança).
4. Para Create Security Group, faça o seguinte:
 - a. Para Security group name, digite um nome para security group que você está criando.
 - b. (Opcional) Digite uma descrição do security group que você está criando.
 - c. Para VPC, escolha a VPC que contém a instância do Amazon EC2 do servidor Web.
 - d. Selecione Adicionar regra.
 - e. Em Tipo, selecione HTTPS na janela suspensa.
 - f. Em Fonte, insira um local de origem.
 - g. Escolha Create security group (Criar grupo de segurança).
5. No painel de navegação, escolha Instâncias.
6. Marque a caixa de seleção ao lado da sua instância do servidor Web.
7. Selecione o menu suspenso Ações, na parte superior da página. Selecione Segurança e, em seguida, Alterar grupos de segurança.
8. Em Grupos de segurança associados, escolha a caixa de pesquisa e escolha o grupo de segurança que você criou para HTTPS. Em seguida, escolha Adicionar grupos de segurança.
9. Selecione Save (Salvar).

Verificar se o HTTPS usa o certificado que você configurou

Depois de adicionar o servidor web a um grupo de segurança, você pode verificar se o SSL/TLS offload está usando seu certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como [OpenSSL s_client](#).

Para verificar o SSL/TLS descarregamento com um navegador da web

1. Use um navegador da Web para se conectar ao servidor Web usando o nome de DNS público ou endereço IP do servidor. Certifique-se de que a URL na barra de endereços comece com `https://`. Por exemplo, `https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, `https://www.example.com/`) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Use seu navegador da Web para ver o certificado do servidor Web. Para obter mais informações, consulte:
 - Para o Mozilla Firefox, consulte o tópico sobre como [Visualizar um certificado](#), no site de suporte da Mozilla.
 - Para o Google Chrome, consulte [Noções básicas de problemas de segurança](#), no site Google Tools for Web Developers.

Outros navegadores da Web podem ter recursos semelhantes, que você pode usar para visualizar o certificado do servidor Web.

3. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Para verificar o SSL/TLS descarregamento com o OpenSSL `s_client`

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS.
`<server name>` Substitua pelo nome DNS público ou endereço IP do seu servidor web.

```
openssl s_client -connect <server name>:443
```


 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Certifique-se de que o SSL/TLS certificado seja aquele que você configurou seu servidor web para usar.

Agora você tem um site protegido com HTTPS. A chave privada do servidor web é armazenada em um HSM no seu AWS CloudHSM cluster.

Para adicionar um balanceador de carga, consulte [Adicione um balanceador de carga com o Elastic Load Balancing AWS CloudHSM para \(opcional\)](#).

Adicione um balanceador de carga com o Elastic Load Balancing AWS CloudHSM para (opcional)

Depois SSL/TLS de configurar o offload com um servidor web, você pode criar mais servidores web e um balanceador de carga do Elastic Load Balancing que roteia o tráfego HTTPS para os servidores web. Um balanceador de carga pode reduzir a carga nos seus servidores Web individuais, equilibrando o tráfego em dois ou mais servidores. Ele também pode aumentar a disponibilidade do seu site, pois o balanceador de carga monitora a integridade dos seus servidores Web e roteia apenas o tráfego aos servidores íntegros. Se um servidor Web falhar, o balanceador de carga deixará automaticamente de rotear o tráfego para ele.

Tópicos

- [Etapa 1. Criar uma sub-rede para um segundo servidor Web](#)
- [Etapa 2. Criar o segundo servidor Web](#)
- [Etapa 3. Criar o balanceador de carga](#)

Etapa 1. Criar uma sub-rede para um segundo servidor Web

Antes de criar outro servidor web, você precisa criar uma nova sub-rede na mesma VPC que contém seu servidor AWS CloudHSM web e cluster existentes.

Para criar uma nova sub-rede

1. Abra a [seção Sub-redes do console Amazon VPC](#).
2. Selecione Create Subnet.
3. Na caixa de diálogo Criar sub-rede, faça o seguinte:
 - a. Em Name tag (Tag de nome), digite um nome para a sub-rede.
 - b. Para VPC, escolha a AWS CloudHSM VPC que contém seu servidor web e cluster existentes. AWS CloudHSM
 - c. Para Availability Zone, escolha uma zona de disponibilidade diferente da que contém seu servidor Web existente.
 - d. Para IPv4 Bloco CIDR, digite o bloco CIDR para uso na sub-rede. Por exemplo, digite **10.0.10.0/24**.
 - e. Escolha Yes, Create (Sim, criar).
4. Marque a caixa de seleção ao lado da sub-rede pública que contém seu servidor Web existente. Ela é diferente da sub-rede pública que você criou na etapa anterior.
5. No painel conteúdo, escolha a guia Tabela de rotas. Em seguida, escolha o link para a tabela de rotas.

subnet-1f358d78 | CloudHSM Public subnet

Summary	Route Table	Network ACL
Edit		
Route Table:		rtb-cea112a9
Destination	Target	
10.0.0.0/16	local	
0.0.0.0/0	igw-68ee440c	

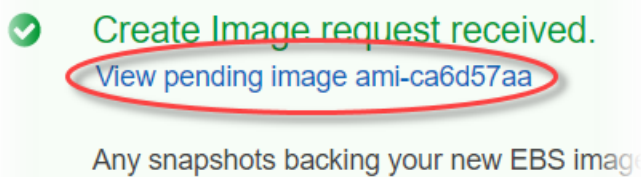
6. Marque a caixa de seleção ao lado da tabela de rotas.
7. Escolha a guia Associações de sub-redes. Em seguida, escolha Editar.
8. Marque a caixa de seleção ao lado da sub-rede pública que você criou anteriormente neste procedimento. Em seguida, escolha Salvar.

Etapa 2. Criar o segundo servidor Web

Complete as seguintes etapas para criar um segundo servidor Web com a mesma configuração que o seu servidor Web existente.

Para criar um segundo servidor Web

1. Abra a seção [Instâncias](#) do console do Amazon EC2 em.
2. Marque a caixa de seleção ao lado da instância do servidor Web existente.
3. Escolha Actions (Ações), Image (Imagem) e, em seguida, Create Image (Criar imagem).
4. Na caixa de diálogo Create Image, faça o seguinte:
 - a. Em Image name (Nome de imagem), digite um nome para a imagem.
 - b. Em Image description (Descrição da imagem), digite uma descrição para a imagem.
 - c. Escolha Create Image. Essa ação reinicia seu servidor Web existente.
 - d. Escolha o **<AMI ID>** link Exibir imagem pendente.



Na coluna Status, anote o status da imagem. Quando o status da imagem for disponível (isso pode demorar vários minutos), vá para a próxima etapa.

5. No painel de navegação, escolha Instâncias.
6. Marque a caixa de seleção ao lado do seu servidor Web existente.
7. Selecione Actions (Ações) e selecione Launch More Like This (Iniciar mais como este).
8. Escolha Edit AMI.

▼ AMI Details



amzn-ami-hvm-2017.09.1.20171120-x86_64-gp2 - ami-a51f27c5

Amazon Linux AMI 2017.09.1.20171120 x86_64 HVM GP2

Root Device Type: ebs Virtualization type: hvm

[Edit AMI](#)

9. No painel de navegação esquerdo, escolha Meu AMIs. Em seguida, limpe o texto na caixa de pesquisa.
10. Ao lado de sua imagem do servidor Web, escolha Select.

11. Escolha Sim, eu quero continuar com essa AMI (**<image name>**- ami-**<AMI ID>**).
12. Escolha Próximo.
13. Selecione um tipo de instância e, a seguir, escolha Próximo: Configurar detalhes da instância.
14. Na Etapa 3: Configurar os detalhes da instância, faça o seguinte:
 - a. Para Network, escolha a VPC que contém seu servidor Web existente.
 - b. Para Subnet, escolha a sub-rede pública que você criou para o segundo servidor Web.
 - c. Para Atribuir IP público automaticamente, selecione Permitir.
 - d. Altere os detalhes restantes da instância conforme preferir. Em seguida, selecione Next: Add Storage (Próximo: adicionar armazenamento).
15. Altere as configurações de armazenamento conforme preferir. Depois, selecione Next: Add Tags (Próximo: adicionar tags).
16. Adicione ou edite tags como preferir. Em seguida, escolha Next: Configure Security Group.
17. Para Etapa 6: Configurar security group, faça o seguinte:
 - a. Em Assign a security group (Atribuir um grupo de segurança), escolha Select an existing security group (Selecionar um security group existente).
 - b. Marque a caixa de seleção ao lado do grupo de segurança chamado cloudhsm- **<cluster ID>** -sg. AWS CloudHSM criou esse grupo de segurança em seu nome quando você [criou o cluster](#). Você deve escolher esse grupo de segurança para permitir que a instância do servidor web se conecte ao HSMs no cluster.
 - c. Marque a caixa de seleção ao lado do grupo de segurança que permite o tráfego HTTPS de entrada. Você [criou esse grupo de segurança anteriormente](#).
 - d. (Opcional) Marque a caixa de seleção ao lado de um grupo de segurança que permite a entrada de tráfego SSH (para Linux) ou RDP (para Windows) da rede. Ou seja, o grupo de segurança deve permitir tráfego TCP de entrada na porta 22 (para SSH no Linux) ou na porta 3389 (para RDP no Windows). Caso contrário, não será possível se conectar à instância do cliente. Caso não tenha um grupo de segurança como esse, crie um e, em seguida, atribua-o para a instância do cliente mais tarde.

Escolha revisar e iniciar.

18. Verifique os detalhes da instância e, em seguida, selecione Iniciar.
19. Escolha se deseja iniciar a instância com um par de chaves existente, criar um novo par de chaves ou executar sua instância sem um par de chaves.

- Para usar um par de chaves existente, faça o seguinte:
 1. Escolha Selecionar um par de chaves existente.
 2. Para Selecionar um par de chaves, selecione o par de chaves a ser usado.
 3. Marque a caixa de seleção ao lado de Eu reconheço que tenho acesso ao arquivo de chave privada selecionado (**<private key file name>.pem**) e que, sem esse arquivo, não conseguirei fazer login na minha instância.
- Para criar um novo par de chaves, faça o seguinte:
 1. Selecione Criar um novo par de chaves.
 2. Em Key pair name (Nome do par de chaves), digite um nome para o par de chaves.
 3. Selecione Fazer o download do par de chaves e salve o arquivo da chave privada em um local seguro e acessível.

**Warning**

Não será possível fazer o download do arquivo de chave privada novamente a partir desse momento. Se você não fizer o download do arquivo de chave privada agora, não será possível acessar a instância do cliente.

- Para iniciar sua instância sem um par de chaves, faça o seguinte:
 1. Escolha Proceed without a key pair.
 2. Marque a caixa de seleção ao lado de I acknowledge that I will not be able to connect to this instance unless I already know the password built into this AMI.

Selecione Launch Instances.

Etapa 3. Criar o balanceador de carga

Conclua as seguintes etapas para criar um balanceador de carga no Elastic Load Balancing que roteia o tráfego HTTPS aos seus servidores Web.

Para criar um balanceador de carga

1. Abra a seção [Balanceadores de carga](#) do console do Amazon EC2.
2. Selecione Criar load balancer.

3. Na seção Network Load Balancer section, escolha Create.
4. Para Step 1: Configure Load Balancer, faça o seguinte:
 - a. Para Name, digite um nome para o load balancer que você está criando.
 - b. Na seção Receptores, para Porta do balanceador de carga, altere o valor para **443**.
 - c. Na seção Availability Zones, para VPC, escolha a VPC que contém seus servidores Web.
 - d. Na seção Availability Zones, escolha as sub-redes que contêm seus servidores Web.
 - e. Selecione Next: Configure Routing (Próximo: Configurar roteamento).
5. Para Step 2: Configure Routing, faça o seguinte:
 - a. Para Name, digite um nome para o grupo-alvo que você está criando.
 - b. Para Porta, altere o valor para **443**.
 - c. Selecione Next: Register Targets (Próximo: Registrar destinos).
6. Para Step 3: Register Targets, faça o seguinte:
 - a. Na seção Instâncias, marque as caixas de seleção ao lado das instâncias do servidor Web. Em seguida, escolha Add to registered.
 - b. Escolha Próximo: revisar.
7. Revise os detalhes do load balancer e escolha Create.
8. Quando o load balancer tiver sido criado com êxito, escolha Close.

Depois de concluir as etapas anteriores, o console do Amazon EC2 mostrará seu balanceador de carga no Elastic Load Balancing.

Quando o estado do balanceador de carga estiver ativo, você poderá verificar se ele está funcionando. Ou seja, você pode verificar se ele está enviando tráfego HTTPS para seus servidores web com SSL/TLS offload with. AWS CloudHSM Faça isso com um navegador da web ou com uma ferramenta como [OpenSSL s_client](#).

Para verificar se o seu load balancer está funcionando com um navegador da Web

1. No console do Amazon EC2, encontre o nome DNS para o balanceador de carga que você acabou de criar. Em seguida, selecione o nome DNS e copie-o.
2. Use um navegador da web, como o Mozilla Firefox ou o Google Chrome, para se conectar ao seu balanceador de carga usando o nome DNS do balanceador de carga. Certifique-se de que a URL na barra de endereços comece com https://.

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

3. Use seu navegador da Web para ver o certificado do servidor Web. Para obter mais informações, consulte:
 - Para o Mozilla Firefox, consulte o tópico sobre como [Visualizar um certificado](#), no site de suporte da Mozilla.
 - Para o Google Chrome, consulte [Noções básicas de problemas de segurança](#), no site Google Tools for Web Developers.

Outros navegadores da Web podem ter recursos semelhantes, que você pode usar para visualizar o certificado do servidor Web.

4. Garanta que o certificado é aquele para o qual você configurou o servidor Web para usar.

Para verificar se o seu load balancer está funcionando com o OpenSSL s_client

1. Use o seguinte comando OpenSSL para se conectar ao seu balanceador de carga usando HTTPS. `<DNS name>` Substitua pelo nome DNS do seu balanceador de carga.

```
openssl s_client -connect <DNS name>:443
```

 Tip

Você pode usar um serviço de DNS, como o Amazon Route 53, para rotear o nome de domínio do seu site (por exemplo, <https://www.example.com/>) para o seu servidor web. Para obter mais informações, consulte [Roteamento de tráfego para uma instância do Amazon EC2](#) no Amazon Route 53 ou na documentação do seu serviço de DNS.

2. Garanta que o certificado é aquele para o qual você configurou o servidor Web para usar.

Agora você tem um site protegido com HTTPS, com a chave privada do servidor web armazenada em um HSM em seu AWS CloudHSM cluster. Seu site possui dois servidores Web e um load balancer para ajudar a melhorar a eficiência e a disponibilidade.

Configure o Windows Server como uma autoridade de certificação (CA) com AWS CloudHSM

AWS CloudHSM oferece suporte para configurar o Windows Server como uma autoridade de certificação (CA) por meio do Client SDK 3 e do Client SDK 5. As etapas para usar essas ferramentas variam de acordo com a versão do Client SDK na qual você baixou atualmente. As seções a seguir fornecem informações sobre cada SDK.

Tópicos

- [Configurar o Windows Server como uma autoridade de certificação \(CA\) com o Client SDK 5](#)
- [Configurar o Windows Server como uma autoridade de certificação \(CA\) com o Client SDK 3](#)

Configurar o Windows Server como uma autoridade de certificação (CA) com o Client SDK 5

Em uma infraestrutura de chave pública (PKI), uma autoridade de certificação (CA) é uma entidade confiável que emite certificados digitais. Esses certificados digitais vinculam uma chave pública a uma identidade (uma pessoa ou organização) por meio de criptografia de chave pública e assinaturas digitais. Para operar uma CA, é necessário manter a confiança, protegendo as chaves privadas que assinam os certificados emitidos pela CA. Armazene essas chaves privadas no HSM em seu cluster AWS CloudHSM e use-o para executar as operações de assinatura criptográfica.

Neste tutorial, você usa o Windows Server e AWS CloudHSM configura uma CA. Instale o software cliente do AWS CloudHSM para Windows no Windows Server e adicione a função Active Directory Certificate Services (AD CS) ao Windows Server. Ao configurar essa função, você usa um provedor de armazenamento de AWS CloudHSM chaves (KSP) para criar e armazenar a chave privada da CA em seu AWS CloudHSM cluster. O KSP é a ponte que conecta o servidor Windows ao AWS CloudHSM cluster. Na última etapa, você assina uma solicitação de assinatura de certificado (CSR) com a CA do Windows Server.

Para saber mais, consulte os seguintes tópicos:

Tópicos

- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: Criar uma CA do Windows Server com AWS CloudHSM](#)
- [Etapa 3: Assine uma solicitação de assinatura de certificado \(CSR\) com sua CA do Windows Server com AWS CloudHSM](#)

Etapa 1: configurar os pré-requisitos

Para configurar o Windows Server como uma autoridade de certificação (CA) com AWS CloudHSM, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos um HSM.
- Uma instância do Amazon EC2 executando um sistema operacional Windows Server com o software AWS CloudHSM cliente para Windows instalado. Este tutorial usa o Microsoft Windows Server 2016.
- Um usuário de criptografia (CU) para possuir e gerenciar a chave privada da CA no HSM.

Para configurar os pré-requisitos para uma CA do Windows Server com AWS CloudHSM

1. Siga as etapas em [Introdução](#). Quando você iniciar o cliente do Amazon EC2, escolha uma AMI do Windows Server. Este tutorial usa o Microsoft Windows Server 2016. Quando você concluir estas etapas, terá um cluster ativo com um HSM, no mínimo. Você também tem uma instância cliente do Amazon EC2 executando o Windows Server com o software AWS CloudHSM cliente para Windows instalado.
2. (Opcional) Adicione mais HSMs ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
3. Conecte-se à instância do cliente. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
4. Crie um usuário de criptografia (CU) usando [Gerenciamento de usuários do HSM com a CloudHSM CLI](#) ou [Gerenciamento de usuários do HSM com o CloudHSM Management Utility \(CMU\)](#). Lembre o nome do usuário e a senha do CU. Você precisará deles para concluir a próxima etapa.
5. [Defina as credenciais de login para o HSM](#), usando o nome de usuário e a senha do CU que você criou na etapa anterior.

- Na etapa 5, se você usou o Gerenciador de Credenciais do Windows para definir as credenciais [psexec.exe](#) do HSM, faça o download SysInternals para executar o seguinte comando como NT Authority\ SYSTEM:

```
psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools\set_cloudhsm_credentials.exe"  
--username <USERNAME> --password <PASSWORD>
```

Substitua <USERNAME> e <PASSWORD> pelas credenciais do HSM.

Para criar uma CA do Windows Server com AWS CloudHSM, acesse [Criar CA do Windows Server](#).

Etapa 2: Criar uma CA do Windows Server com AWS CloudHSM

Para criar um CA do Windows Server, adicione a função Active Directory Certificate Services (AD CS) ao Windows Server. Ao adicionar essa função, você usa um provedor de armazenamento de AWS CloudHSM chaves (KSP) para criar e armazenar a chave privada da CA em seu AWS CloudHSM cluster.

Note

Ao criar a CA do Windows Server, você pode optar por criar uma CA raiz ou uma CA subordinada. Você normalmente toma essa decisão com base no design da sua infraestrutura de chave pública e nas políticas de segurança da sua organização. Este tutorial explica como criar uma CA raiz para simplificar.

Para adicionar a função AD CS ao Windows Server e criar a chave privada da CA

- Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
- No Windows Server, inicie o Gerenciador de servidores.
- No painel Server Manager, escolha Add roles and features.
- Leia as informações de Before you begin e escolha Next.
- Em Installation Type (Tipo de instalação), escolha Role-based or feature-based installation (Instalação com base na função ou no recurso). Em seguida, escolha Next (Próximo).
- Em Server Selection, escolha Select a server from the server pool. Escolha Próximo.
- Em Server Roles, faça o seguinte:

- a. Selecione Active Directory Certificate Services.
 - b. Em Add features that are required for Active Directory Certificate Services, escolha Add Features.
 - c. Escolha Next para terminar de selecionar funções de servidor.
8. Em Recursos, aceite os padrões e escolha Next.
 9. Em AD CS, faça o seguinte:
 - a. Escolha Próximo.
 - b. Selecione Certification Authority e escolha Next.
 10. Em Confirmation, leia as informações de confirmação e escolha Install. Não feche a janela.
 11. Escolha o link destacado Configurar Active Directory Certificate Services no servidor de destino.
 12. Em Credentials, verifique ou altere as credenciais exibidas. Escolha Próximo.
 13. Em Role Services, selecione Certification Authority. Escolha Próximo.
 14. Em Setup Type, selecione Standalone CA. Escolha Próximo.
 15. Em CA Type, selecione Root CA. Escolha Próximo.

 Note

Você pode optar por criar uma CA raiz ou uma CA subordinada com base no design da sua infraestrutura de chave pública e nas políticas de segurança da sua organização. Este tutorial explica como criar uma CA raiz para simplificar.

16. Em Private Key, selecione Create a new private key. Escolha Próximo.
17. Em Cryptography, faça o seguinte:
 - a. Em Selecionar um provedor criptográfico, escolha uma das opções de CloudHSM Key Storage Provider no menu. Esses são os provedores de armazenamento de chaves do AWS CloudHSM . Por exemplo, você pode escolher RSA#CloudHSM Key Storage Provider.
 - b. Em Key length, escolha uma das opções de tamanho de chave.
 - c. Em Select the hash algorithm for signing certificates issued by this CA, escolha uma das opções de algoritmo hash.

Escolha Próximo.

18. Em CA Name, faça o seguinte:

- a. (Opcional) Edite o nome comum.
- b. (Opcional) Digite um sufixo de nome distinto.

Escolha Próximo.

19. Em Validity Period, especifique um período em anos, meses, semanas ou dias. Escolha Próximo.
20. Em Certificate Database, aceite os valores padrão ou, se desejar, altere o local do banco de dados e do log do banco de dados. Escolha Próximo.
21. Em Confirmation, analise as informações sobre a CA e escolha Configure.
22. Escolha Close e, em seguida, escolha Close novamente.

Agora você tem uma CA do Windows Server com AWS CloudHSM. Para saber como assinar uma solicitação de assinatura de certificado (CSR) com a CA, vá para [Assine uma CSR](#).

Etapa 3: Assine uma solicitação de assinatura de certificado (CSR) com sua CA do Windows Server com AWS CloudHSM

Você pode usar sua CA do Windows Server com AWS CloudHSM para assinar uma solicitação de assinatura de certificado (CSR). Para concluir estas etapas, você precisa de uma CSR válida. Você pode criar uma CSR de várias formas:

- Usando o OpenSSL
- Usando o Gerenciador do Serviços de Informações da Internet (IIS) do Windows Server
- Usando o snap-in de certificados no Microsoft Management Console
- Usando o utilitário de linha de comando certreq no Windows

As etapas de criação de uma CSR estão fora do escopo deste tutorial. Quando você tiver uma CSR, poderá assiná-lo com a CA do Windows Server.

Para assinar uma CSR com a CA do Windows Server

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
2. No Windows Server, inicie o Gerenciador de servidores.

3. No painel Server Manager, no canto superior direito, escolha Tools, Certification Authority.
4. Na janela Autoridade de Certificação, escolha o nome do computador.
5. No menu Action, escolha All Tasks, Submit new request.
6. Selecione o arquivo CSR e escolha Open.
7. Na janela Certification Authority, clique duas vezes em Pending Requests.
8. Selecione a solicitação pendente. No menu Action, escolha All Tasks, Issue.
9. Na janela Certification Authority, clique duas vezes em Issued Requests para visualizar o certificado assinado.
10. (Opcional) Para exportar o certificado assinado para um arquivo, execute as seguintes etapas:
 - a. Na janela Certification Authority, clique duas vezes no certificado.
 - b. Escolha a guia Details e escolha Copy to File.
 - c. Siga as instruções no Certificate Export Wizard.

Agora você tem uma CA do Windows Server com AWS CloudHSM, e um certificado válido assinado pela CA do Windows Server.

Configurar o Windows Server como uma autoridade de certificação (CA) com o Client SDK 3

Em uma infraestrutura de chave pública (PKI), uma autoridade de certificação (CA) é uma entidade confiável que emite certificados digitais. Esses certificados digitais vinculam uma chave pública a uma identidade (uma pessoa ou organização) por meio de criptografia de chave pública e assinaturas digitais. Para operar uma CA, é necessário manter a confiança, protegendo as chaves privadas que assinam os certificados emitidos pela CA. Armazene essas chaves privadas no HSM em seu cluster AWS CloudHSM e use-o para executar as operações de assinatura criptográfica.

Neste tutorial, você usa o Windows Server e AWS CloudHSM configura uma CA. Instale o software cliente do AWS CloudHSM para Windows no Windows Server e adicione a função Active Directory Certificate Services (AD CS) ao Windows Server. Ao configurar essa função, você usa um provedor de armazenamento de AWS CloudHSM chaves (KSP) para criar e armazenar a chave privada da CA em seu AWS CloudHSM cluster. O KSP é a ponte que conecta o servidor Windows ao AWS CloudHSM cluster. Na última etapa, você assina uma solicitação de assinatura de certificado (CSR) com a CA do Windows Server.

Para saber mais, consulte os seguintes tópicos:

Tópicos

- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: Criar uma CA do Windows Server com AWS CloudHSM](#)
- [Etapa 3: Assine uma solicitação de assinatura de certificado \(CSR\) com sua CA do Windows Server com AWS CloudHSM](#)

Etapa 1: configurar os pré-requisitos

Para configurar o Windows Server como uma autoridade de certificação (CA) com AWS CloudHSM, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos um HSM.
- Uma instância do Amazon EC2 executando um sistema operacional Windows Server com o software AWS CloudHSM cliente para Windows instalado. Este tutorial usa o Microsoft Windows Server 2016.
- Um usuário de criptografia (CU) para possuir e gerenciar a chave privada da CA no HSM.

Para configurar os pré-requisitos para uma CA do Windows Server com AWS CloudHSM

1. Siga as etapas em [Introdução](#). Quando você iniciar o cliente do Amazon EC2, escolha uma AMI do Windows Server. Este tutorial usa o Microsoft Windows Server 2016. Quando você concluir estas etapas, terá um cluster ativo com um HSM, no mínimo. Você também tem uma instância cliente do Amazon EC2 executando o Windows Server com o software AWS CloudHSM cliente para Windows instalado.
2. (Opcional) Adicione mais HSMs ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
3. Conecte-se à instância do cliente. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
4. Crie um usuário de criptografia (CU) usando [Gerenciamento de usuários do HSM com a CloudHSM CLI](#) ou [Gerenciamento de usuários do HSM com o CloudHSM Management Utility \(CMU\)](#). Lembre o nome do usuário e a senha do CU. Você precisará deles para concluir a próxima etapa.
5. [Defina as credenciais de login para o HSM](#), usando o nome de usuário e a senha do CU que você criou na etapa anterior.

- Na etapa 5, se você usou o Gerenciador de Credenciais do Windows para definir as credenciais [psexec.exe](#) do HSM, faça o download SysInternals para executar o seguinte comando como NT Authority\ SYSTEM:

```
psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools  
\set_cloudhsm_credentials.exe" --username <USERNAME> --password <PASSWORD>
```

Substitua <USERNAME> e <PASSWORD> pelas credenciais do HSM.

Para criar uma CA do Windows Server com AWS CloudHSM, acesse [Criar CA do Windows Server](#).

Etapa 2: Criar uma CA do Windows Server com AWS CloudHSM

Para criar um CA do Windows Server, adicione a função Active Directory Certificate Services (AD CS) ao Windows Server. Ao adicionar essa função, você usa um provedor de armazenamento de AWS CloudHSM chaves (KSP) para criar e armazenar a chave privada da CA em seu AWS CloudHSM cluster.

Note

Ao criar a CA do Windows Server, você pode optar por criar uma CA raiz ou uma CA subordinada. Você normalmente toma essa decisão com base no design da sua infraestrutura de chave pública e nas políticas de segurança da sua organização. Este tutorial explica como criar uma CA raiz para simplificar.

Para adicionar a função AD CS ao Windows Server e criar a chave privada da CA

- Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
- No Windows Server, inicie o Gerenciador de servidores.
- No painel Server Manager, escolha Add roles and features.
- Leia as informações de Before you begin e escolha Next.
- Em Installation Type (Tipo de instalação), escolha Role-based or feature-based installation (Instalação com base na função ou no recurso). Em seguida, escolha Next (Próximo).
- Em Server Selection, escolha Select a server from the server pool. Escolha Próximo.

7. Em Server Roles, faça o seguinte:
 - a. Selecione Active Directory Certificate Services.
 - b. Em Add features that are required for Active Directory Certificate Services, escolha Add Features.
 - c. Escolha Next para terminar de selecionar funções de servidor.
8. Em Recursos, aceite os padrões e escolha Next.
9. Em AD CS, faça o seguinte:
 - a. Escolha Próximo.
 - b. Selecione Certification Authority e escolha Next.
10. Em Confirmation, leia as informações de confirmação e escolha Install. Não feche a janela.
11. Escolha o link destacado Configurar Active Directory Certificate Services no servidor de destino.
12. Em Credentials, verifique ou altere as credenciais exibidas. Escolha Próximo.
13. Em Role Services, selecione Certification Authority. Escolha Próximo.
14. Em Setup Type, selecione Standalone CA. Escolha Próximo.
15. Em CA Type, selecione Root CA. Escolha Próximo.

 Note

Você pode optar por criar uma CA raiz ou uma CA subordinada com base no design da sua infraestrutura de chave pública e nas políticas de segurança da sua organização. Este tutorial explica como criar uma CA raiz para simplificar.

16. Em Private Key, selecione Create a new private key. Escolha Próximo.
17. Em Cryptography, faça o seguinte:
 - a. Em Select a cryptographic provider, escolha uma das opções de Cavium Key Storage Provider no menu. Esses são os provedores de armazenamento de chaves do AWS CloudHSM . Por exemplo, você pode escolher RSA#Cavium Key Storage Provider.
 - b. Em Key length, escolha uma das opções de tamanho de chave.
 - c. Em Select the hash algorithm for signing certificates issued by this CA, escolha uma das opções de algoritmo hash.

Escolha Próximo.

18. Em CA Name, faça o seguinte:

- a. (Opcional) Edite o nome comum.
- b. (Opcional) Digite um sufixo de nome distinto.

Escolha Próximo.

19. Em Validity Period, especifique um período em anos, meses, semanas ou dias. Escolha Próximo.

20. Em Certificate Database, aceite os valores padrão ou, se desejar, altere o local do banco de dados e do log do banco de dados. Escolha Próximo.

21. Em Confirmation, analise as informações sobre a CA e escolha Configure.

22. Escolha Close e, em seguida, escolha Close novamente.

Agora você tem uma CA do Windows Server com AWS CloudHSM. Para saber como assinar uma solicitação de assinatura de certificado (CSR) com a CA, vá para [Assine uma CSR](#).

Etapa 3: Assine uma solicitação de assinatura de certificado (CSR) com sua CA do Windows Server com AWS CloudHSM

Você pode usar sua CA do Windows Server com AWS CloudHSM para assinar uma solicitação de assinatura de certificado (CSR). Para concluir estas etapas, você precisa de uma CSR válida. Você pode criar uma CSR de várias formas:

- Usando o OpenSSL
- Usando o Gerenciador do Serviços de Informações da Internet (IIS) do Windows Server
- Usando o snap-in de certificados no Microsoft Management Console
- Usando o utilitário de linha de comando certreq no Windows

As etapas de criação de uma CSR estão fora do escopo deste tutorial. Quando você tiver uma CSR, poderá assiná-lo com a CA do Windows Server.

Para assinar uma CSR com a CA do Windows Server

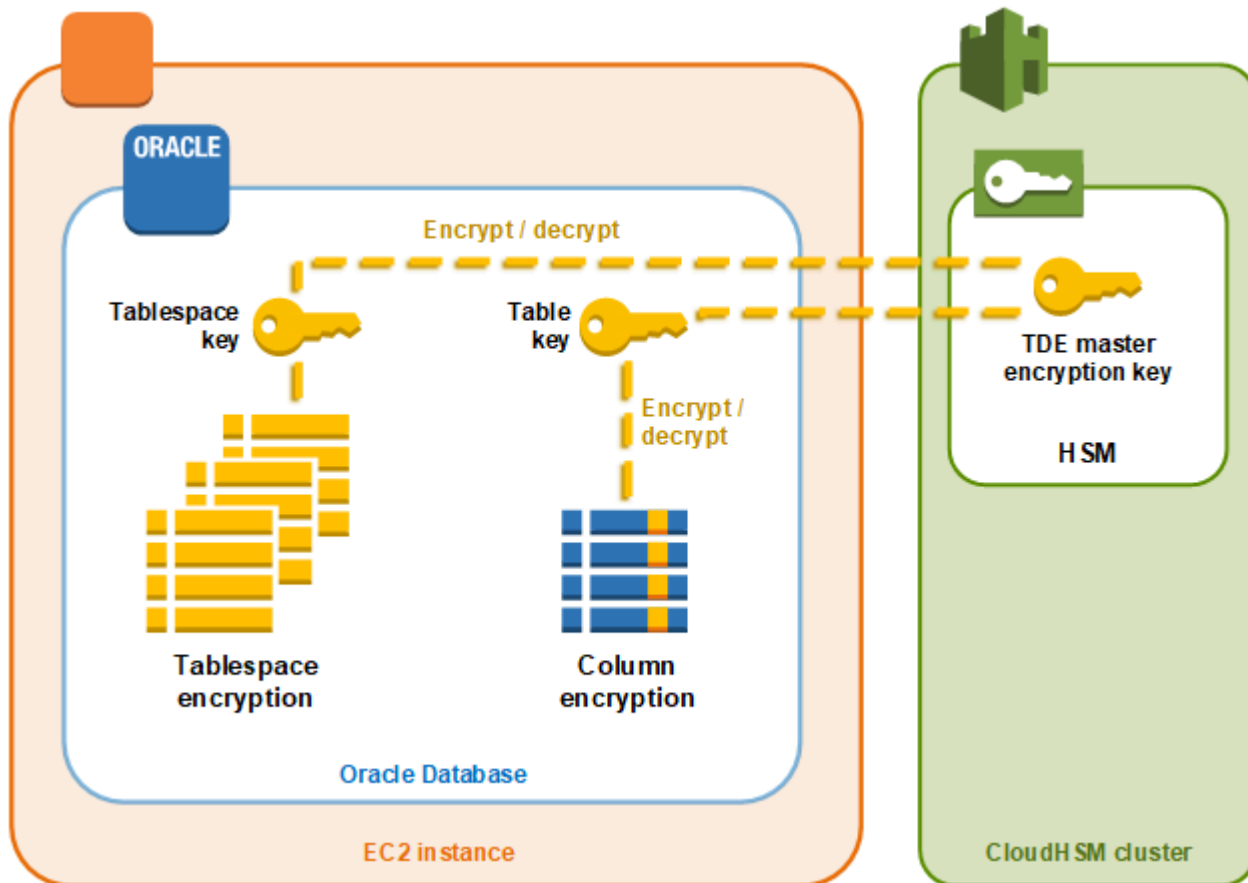
1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.

2. No Windows Server, inicie o Gerenciador de servidores.
3. No painel Server Manager, no canto superior direito, escolha Tools, Certification Authority.
4. Na janela Autoridade de Certificação, escolha o nome do computador.
5. No menu Action, escolha All Tasks, Submit new request.
6. Selecione o arquivo CSR e escolha Open.
7. Na janela Certification Authority, clique duas vezes em Pending Requests.
8. Selecione a solicitação pendente. No menu Action, escolha All Tasks, Issue.
9. Na janela Certification Authority, clique duas vezes em Issued Requests para visualizar o certificado assinado.
10. (Opcional) Para exportar o certificado assinado para um arquivo, execute as seguintes etapas:
 - a. Na janela Certification Authority, clique duas vezes no certificado.
 - b. Escolha a guia Details e escolha Copy to File.
 - c. Siga as instruções no Certificate Export Wizard.

Agora você tem uma CA do Windows Server com AWS CloudHSM, e um certificado válido assinado pela CA do Windows Server.

Criptografia de dados transparente (TDE) do banco de dados Oracle com AWS CloudHSM

Transparent Data Encryption [criptografia de dados transparente (TDE)] é usada para criptografar arquivos de banco de dados. Com o TDE, o servidor de banco de dados criptografa os dados antes de armazená-los no disco. Os dados nas colunas ou tablespaces da tabela do banco de dados são criptografados com uma chave de tabela ou chave de tablespace. Algumas versões do software de banco de dados da Oracle oferecem TDE. No Oracle TDE, essas chaves são criptografadas com a chave mestre de criptografia TDE. Você pode obter maior segurança armazenando a chave mestra de criptografia do TDE HSMs no seu AWS CloudHSM cluster.



Nesta solução, você usa o Oracle Database instalado em uma instância do Amazon EC2. O Oracle Database se integra à [biblioteca de AWS CloudHSM software do PKCS #11](#) para armazenar a chave mestra do TDE HSMs no seu cluster.

⚠ Important

- Recomendamos instalar o Oracle Database em uma instância do Amazon EC2.

Complete as seguintes etapas para realizar a integração do Oracle TDE com o AWS CloudHSM.

Tópicos

- [Etapa 1. Configurar os pré-requisitos](#)
- [Etapa 3: gerar a chave de criptografia mestra do Oracle TDE](#)

Etapa 1. Configurar os pré-requisitos

Para realizar a integração do Oracle TDE com AWS CloudHSM, você precisa do seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos um HSM.
- Uma instância do Amazon EC2 executando um sistema operacional Amazon Linux com o seguinte software instalado:
 - O AWS CloudHSM cliente e as ferramentas da linha de comando.
 - A biblioteca AWS CloudHSM de software para PKCS #11.
 - Banco de dados Oracle. AWS CloudHSM oferece suporte à integração com o Oracle TDE. O Client SDK 5.6 e versões superiores oferecem suporte ao Oracle TDE para o Oracle Database 19c. O Client SDK 3 oferece suporte ao Oracle TDE para o Oracle Database versões 11g e 12c.
- Um usuário criptográfico (UC) para possuir e gerenciar a chave de criptografia mestra do TDE HSMs no seu cluster.

Conclua as seguintes etapas para configurar todos os pré-requisitos.

Para configurar os pré-requisitos para a integração do Oracle TDE com AWS CloudHSM

1. Siga as etapas em [Introdução](#). Depois de fazer isso, você terá um cluster ativo com um HSM. Você também terá uma instância do Amazon EC2 executando o sistema operacional Amazon Linux. As ferramentas AWS CloudHSM do cliente e da linha de comando também serão instaladas e configuradas.
2. (Opcional) Adicione mais HSMs ao seu cluster. Para obter mais informações, consulte [Adicionar um HSM a um cluster AWS CloudHSM](#).
3. Conecte-se à sua instância de cliente do Amazon EC2 e faça o seguinte:
 - a. [Instale a biblioteca AWS CloudHSM de software para PKCS #11](#).
 - b. Instale o Oracle Database. Para obter mais informações, consulte a [documentação do Oracle Database](#). O Client SDK 5.6 e versões superiores oferecem suporte ao Oracle TDE para o Oracle Database 19c. O Client SDK 3 oferece suporte ao Oracle TDE para o Oracle Database versões 11g e 12c.
 - c. Use a ferramenta da linha de comando `cloudhsm_mgmt_util` para criar um usuário de criptografia (CU) no seu cluster. Para obter mais informações sobre a criação de um CU, consulte [Como gerenciar usuários do HSM com CMU](#) e [Usuários do HSM](#).

Etapa 3: gerar a chave de criptografia mestra do Oracle TDE

Para gerar a chave mestra do Oracle TDE HSMs no seu cluster, conclua as etapas do procedimento a seguir.

Para gerar a chave-mestra

1. Use o comando a seguir para abrir o Oracle SQL*Plus. Quando solicitado, digite a senha do sistema que você definiu quando instalou o Oracle Database.

```
sqlplus / as sysdba
```

Note

Para Client SDK 3, é necessário definir a variável de ambiente `CLOUDHSM_IGNORE_CKA_MODIFIABLE_FALSE` sempre que você gerar uma chave mestra. Essa variável é necessária apenas para a geração de chave mestra. Para obter mais informações, consulte "Problema: Oracle define o atributo PKCS #11 `CKA_MODIFIABLE` durante a geração da chave mestra, mas o HSM não oferece suporte a ele" em [Problemas conhecidos para integração de aplicativos de terceiros](#).

2. Execute a instrução SQL que cria a chave de criptografia mestra, conforme mostrado nos exemplos a seguir. Use a instrução que corresponda à sua versão do Oracle Database.

<CU user name> Substitua pelo nome de usuário do usuário criptográfico (UC).

<password> Substitua pela senha da UC.

Important

Execute o seguinte comando apenas uma vez. Cada vez que o comando é executado, ele cria uma nova chave de criptografia mestra.

- Para o Oracle Database versão 11, execute a seguinte instrução SQL.

```
SQL> alter system set encryption key identified by "<CU user name>:<password>";
```

- Para o Oracle Database versão 12 e versão 19c, execute a seguinte instrução SQL.

```
SQL> administer key management set key identified by "<CU user  
name>:<password>";
```

Se a resposta for `System altered` ou `keystore altered`, você gerou e definiu com êxito a chave-mestra do Oracle TDE.

3. (Opcional) Execute o seguinte comando para verificar o status da Oracle wallet.

```
SQL> select * from v$encryption_wallet;
```

Se a carteira não estiver aberta, use um dos seguintes comandos para abri-lo. *<CU user name>* Substitua pelo nome do usuário criptográfico (UC). *<password>* Substitua pela senha da UC.

- Para o Oracle 11, execute o seguinte comando para abrir a wallet.

```
SQL> alter system set encryption wallet open identified by "<CU user  
name>:<password>";
```

Para fechar manualmente a carteira, execute o seguinte comando.

```
SQL> alter system set encryption wallet close identified by "<CU user  
name>:<password>";
```

- Para o Oracle 12 e Oracle 19c, execute o seguinte comando para abrir a wallet.

```
SQL> administer key management set keystore open identified by "<CU user  
name>:<password>";
```

Para fechar manualmente a carteira, execute o seguinte comando.

```
SQL> administer key management set keystore close identified by "<CU user  
name>:<password>";
```

Use o Microsoft SignTool com AWS CloudHSM para assinar arquivos

AWS CloudHSM oferece suporte para usar o Microsoft Signtool para assinar arquivos por meio do Client SDK 3 e do Client SDK 5. As etapas para usar essas ferramentas variam de acordo com a versão do Client SDK na qual você baixou atualmente. As seções a seguir fornecem informações sobre cada SDK.

Tópicos

- [Use a Microsoft SignTool com o Client SDK 5 para assinar arquivos](#)
- [Use a Microsoft SignTool com o Client SDK 3 para assinar arquivos](#)

Use a Microsoft SignTool com o Client SDK 5 para assinar arquivos

Em criptografia e na infraestrutura de chave pública (PKI), as assinaturas digitais são usadas para confirmar que os dados foram enviados por uma entidade confiável. As assinaturas também indicam que os dados não foram alterados em trânsito. Uma assinatura é um hash criptografado que é gerado com a chave privada do remetente. O destinatário pode verificar a integridade dos dados descriptografando sua assinatura de hash com a chave pública do remetente. Por outro lado, o remetente é responsável por manter um certificado digital. O certificado digital demonstra a propriedade da chave privada do remetente e fornece ao destinatário a chave pública que é necessária para a descriptografia. Desde que a chave privada seja de propriedade do remetente, a assinatura pode ser confiável. AWS CloudHSM fornece hardware seguro validado pelo FIPS 140-2 nível 3 para você proteger essas chaves com acesso exclusivo de um único inquilino.

Muitas organizações usam a Microsoft SignTool, uma ferramenta de linha de comando que assina, verifica e registra a data e hora dos arquivos para simplificar o processo de assinatura de código. Você pode usar AWS CloudHSM para armazenar com segurança seus pares de chaves até que sejam necessários SignTool, criando assim um fluxo de trabalho facilmente automatizável para assinar dados.

Os tópicos a seguir fornecem uma visão geral de como usar SignTool com AWS CloudHSM.

Tópicos

- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: criar um certificado de assinatura](#)

- [Etapa 3: assinar um arquivo](#)

Etapa 1: configurar os pré-requisitos

Para usar o Microsoft SignTool com AWS CloudHSM, você precisa do seguinte:

- Uma instância de cliente do Amazon EC2 em execução em um sistema operacional Windows.
- Uma autoridade de certificação (CA), auto-mantida ou estabelecida por um provedor de terceiros.
- Um AWS CloudHSM cluster ativo na mesma nuvem pública virtual (VPC) da sua instância do EC2. O cluster deve conter pelo menos um HSM.
- Um usuário criptográfico (UC) para possuir e gerenciar chaves no AWS CloudHSM cluster.
- Um arquivo não assinado ou executável.
- O Kit de desenvolvimento de software (SDK) do Microsoft Windows.

Para configurar os pré-requisitos para uso com o Windows AWS CloudHSM SignTool

1. Siga as instruções na seção [Getting Started \(Conceitos básicos\)](#) deste guia para executar uma instância do EC2 do Windows e um cluster do AWS CloudHSM .
2. Se você quiser hospedar sua própria CA do Windows Server, siga as etapas 1 e 2 em [Configurando o Windows Server como uma autoridade de certificação com AWS CloudHSM](#). Caso contrário, continue a usar sua CA terceirizada confiável publicamente.
3. Faça download e instale uma das seguintes versões do SDK do Microsoft Windows em sua instância do EC2 do Windows:
 - [SDK 10 do Microsoft Windows](#)
 - [SDK 8.1 do Microsoft Windows](#)
 - [SDK 7 do Microsoft Windows](#)

O executável SignTool faz parte do recurso de instalação das Ferramentas de assinatura do SDK do Windows para aplicativos de área de trabalho. Você pode omitir outros recursos a serem instalados se não forem necessários. O local de instalação padrão é:

```
C:\Program Files (x86)\Windows Kits\<SDK version>\bin\<version number>\<CPU architecture>\signtool.exe
```


Agora você pode usar o SDK do Microsoft Windows, seu AWS CloudHSM cluster e sua CA para [criar um certificado de assinatura](#).

Etapa 2: criar um certificado de assinatura

Agora que você fez download do SDK do Windows para sua instância do EC2, você pode usá-lo para gerar uma solicitação de assinatura de certificado (CSR). A CSR é um certificado não assinado passado para a CA para assinatura. Neste exemplo, usamos o executável `certreq` incluído com o Windows SDK para gerar a CSR.

Para gerar uma CSR usando o executável **certreq**

1. Se ainda não tiver feito isso, conecte-se à instância do EC2 do Windows. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.
2. Crie um arquivo chamado `request.inf` que contenha as linhas a seguir. Substitua as informações de Subject com as de sua organização. Para obter uma explicação de cada parâmetro, consulte a [Documentação da Microsoft](#).

```
[Version]
Signature= $Windows NT$
[NewRequest]
Subject = "C=<Country>,CN=<www.website.com>,O=<Organization>,OU=<Organizational-Unit>,L=<City>,S=<State>"
RequestType=PKCS10
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "CloudHSM Key Storage Provider"
KeyUsage = "CERT_DIGITAL_SIGNATURE_KEY_USAGE"
MachineKeySet = True
Exportable = False
```

3. Executar `certreq.exe`. Para este exemplo, salve a CSR como `request.csr`.

```
certreq.exe -new request.inf request.csr
```

Internamente, um novo par de chaves é gerado em seu AWS CloudHSM cluster e a chave privada do par é usada para criar a CSR.

4. Envie a CSR à sua CA. Se estiver usando uma CA do Windows Server, siga estas etapas:

- a. Digite o seguinte comando para abrir a ferramenta CA:

```
certsrv.msc
```

- b. Na nova janela, clique com o botão direito do mouse no nome do servidor da CA. Escolha All Tasks (Todas as tarefas) e depois escolha Submit new request (Enviar nova solicitação).
- c. Navegue até o local do request.csre escolha Open (Abrir).
- d. Navegue até a pasta Pending Requests (Solicitações pendentes) expandindo o menu Server CA (CA do servidor). Clique com o botão direito do mouse na solicitação que você acabou de criar e, em All Tasks (Todas as tarefas), escolha Issue (Problema).
- e. Agora, navegue até a pasta Issued Certificates (Certificados emitidos) (acima da pasta Pending Requests (Solicitações pendentes)).
- f. Escolha Open (Abrir) para visualizar o certificado e depois escolha a guia Details (Detalhes).
- g. Escolha Copy to File (Copiar no arquivo) para iniciar o Certificate Export Wizard (Assistente de exportação da CA). Salve o arquivo X.509 codificado por DER em um local seguro como signedCertificate.cer.
- h. Saia da ferramenta CA e use o comando a seguir, que move o arquivo do certificado para o armazenamento de certificados pessoais no Windows. Ele pode então ser usado por outras aplicações.

```
certreq.exe -accept signedCertificate.cer
```

Agora, você pode usar o certificado importado para [Assinar um arquivo](#).

Etapa 3: assinar um arquivo

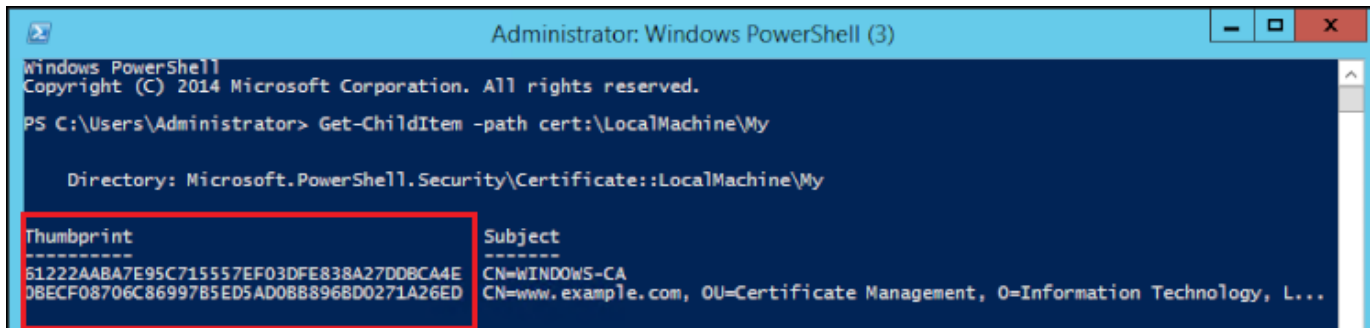
Agora você está pronto para usar SignTool seu certificado importado para assinar seu arquivo de exemplo. Para fazer isso, você precisa saber o hash SHA-1 do certificado ou a impressão digital. A impressão digital é usada para garantir que use SignTool apenas certificados verificados por. AWS CloudHSM Neste exemplo, usamos PowerShell para obter o hash do certificado. Você também pode usar a GUI da CA ou o executável certutil do SDK do Windows.

Para obter uma impressão digital do certificado e usá-lo para assinar um arquivo

1. Abra PowerShell como administrador e execute o seguinte comando:

```
Get-ChildItem -path cert:\LocalMachine\My
```

Copie a Thumbprint que é retornada.



```

Administrator: Windows PowerShell (3)
Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

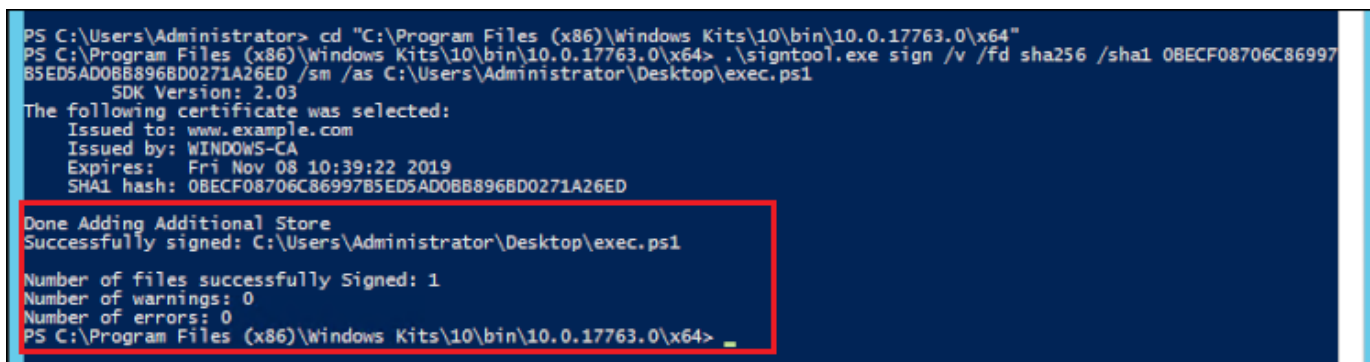
PS C:\Users\Administrator> Get-ChildItem -path cert:\LocalMachine\My

Directory: Microsoft.PowerShell.Security\Certificate::LocalMachine\My

Thumbprint                                     Subject
-----
61222AABA7E95C715557EF03DFE838A27D08CA4E    CN=WINDOWS-CA
0BECF08706C86997B5ED5AD0BB8968D0271A26ED    CN=www.example.com, OU=Certificate Management, O=Information Technology, L...
  
```

2. Navegue até o diretório PowerShell que contém `SignTool.exe`. O local padrão é `C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64`.
3. Finalmente, assine seu arquivo executando o comando a seguir. Se o comando for bem-sucedido, PowerShell retornará uma mensagem de sucesso.

```
signtool.exe sign /v /fd sha256 /sha1 <thumbprint> /sm C:\Users\Administrator\
\Desktop\<test>.ps1
```



```

PS C:\Users\Administrator> cd "C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64"
PS C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64> .\signtool.exe sign /v /fd sha256 /sha1 0BECF08706C86997
B5ED5AD0BB8968D0271A26ED /sm /as C:\Users\Administrator\Desktop\exec.ps1
SDK Version: 2.03
The following certificate was selected:
  Issued to: www.example.com
  Issued by: WINDOWS-CA
  Expires:  Fri Nov 08 10:39:22 2019
  SHA1 hash: 0BECF08706C86997B5ED5AD0BB8968D0271A26ED

Done Adding Additional Store
Successfully signed: C:\Users\Administrator\Desktop\exec.ps1

Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
PS C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64>
  
```

4. (Opcional) Para verificar a assinatura no arquivo, use o seguinte comando:

```
signtool.exe verify /v /pa C:\Users\Administrator\Desktop\<test>.ps1
```

Use a Microsoft SignTool com o Client SDK 3 para assinar arquivos

Em criptografia e na infraestrutura de chave pública (PKI), as assinaturas digitais são usadas para confirmar que os dados foram enviados por uma entidade confiável. As assinaturas também

indicam que os dados não foram alterados em trânsito. Uma assinatura é um hash criptografado que é gerado com a chave privada do remetente. O destinatário pode verificar a integridade dos dados descriptografando sua assinatura de hash com a chave pública do remetente. Por outro lado, o remetente é responsável por manter um certificado digital. O certificado digital demonstra a propriedade da chave privada do remetente e fornece ao destinatário a chave pública que é necessária para a descriptografia. Desde que a chave privada seja de propriedade do remetente, a assinatura pode ser confiável. AWS CloudHSM fornece hardware seguro validado pelo FIPS 140-2 nível 3 para você proteger essas chaves com acesso exclusivo de um único inquilino.

Muitas organizações usam a Microsoft SignTool, uma ferramenta de linha de comando que assina, verifica e registra a data e hora dos arquivos para simplificar o processo de assinatura de código. Você pode usar AWS CloudHSM para armazenar com segurança seus pares de chaves até que sejam necessários SignTool, criando assim um fluxo de trabalho facilmente automatizável para assinar dados.

Os tópicos a seguir fornecem uma visão geral de como usar SignTool com AWS CloudHSM.

Tópicos

- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: criar um certificado de assinatura](#)
- [Etapa 3: assinar um arquivo](#)

Etapa 1: configurar os pré-requisitos

Para usar o Microsoft SignTool com AWS CloudHSM, você precisa do seguinte:

- Uma instância de cliente do Amazon EC2 em execução em um sistema operacional Windows.
- Uma autoridade de certificação (CA), auto-mantida ou estabelecida por um provedor de terceiros.
- Um AWS CloudHSM cluster ativo na mesma nuvem pública virtual (VPC) da sua instância do EC2. O cluster deve conter pelo menos um HSM.
- Um usuário criptográfico (UC) para possuir e gerenciar chaves no AWS CloudHSM cluster.
- Um arquivo não assinado ou executável.
- O Kit de desenvolvimento de software (SDK) do Microsoft Windows.

Para configurar os pré-requisitos para uso com o Windows AWS CloudHSM SignTool

1. Siga as instruções na seção [Getting Started \(Conceitos básicos\)](#) deste guia para executar uma instância do EC2 do Windows e um cluster do AWS CloudHSM .
2. Se você quiser hospedar sua própria CA do Windows Server, siga as etapas 1 e 2 em [Configurando o Windows Server como uma autoridade de certificação com AWS CloudHSM](#). Caso contrário, continue a usar sua CA terceirizada confiável publicamente.
3. Faça download e instale uma das seguintes versões do SDK do Microsoft Windows em sua instância do EC2 do Windows:
 - [SDK 10 do Microsoft Windows](#)
 - [SDK 8.1 do Microsoft Windows](#)
 - [SDK 7 do Microsoft Windows](#)

O executável SignTool faz parte do recurso de instalação das Ferramentas de assinatura do SDK do Windows para aplicativos de área de trabalho. Você pode omitir outros recursos a serem instalados se não forem necessários. O local de instalação padrão é:

```
C:\Program Files (x86)\Windows Kits\<SDK version>\bin\<version number>\<CPU architecture>\signtool.exe
```

Agora você pode usar o SDK do Microsoft Windows, seu AWS CloudHSM cluster e sua CA para [criar um certificado de assinatura](#).

Etapa 2: criar um certificado de assinatura

Agora que você fez download do SDK do Windows para sua instância do EC2, você pode usá-lo para gerar uma solicitação de assinatura de certificado (CSR). A CSR é um certificado não assinado passado para a CA para assinatura. Neste exemplo, usamos o executável `certreq` incluído com o Windows SDK para gerar a CSR.

Para gerar uma CSR usando o executável **certreq**

1. Se ainda não tiver feito isso, conecte-se à instância do EC2 do Windows. Para obter mais informações, consulte [Conectar-se à sua instância](#) no Guia do usuário do Amazon EC2.

2. Crie um arquivo chamado `request.inf` que contenha as linhas a seguir. Substitua as informações de `Subject` com as de sua organização. Para obter uma explicação de cada parâmetro, consulte a [Documentação da Microsoft](#).

```
[Version]
Signature= $Windows NT$
[NewRequest]
Subject = "C=<Country>,CN=<www.website.com>,O=<Organization>,OU=<Organizational-Unit>,L=<City>,S=<State>"
RequestType=PKCS10
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "Cavium Key Storage Provider"
KeyUsage = "CERT_DIGITAL_SIGNATURE_KEY_USAGE"
MachineKeySet = True
Exportable = False
```

3. Executar `certreq.exe`. Para este exemplo, salve a CSR como `request.csr`.

```
certreq.exe -new request.inf request.csr
```

Internamente, um novo par de chaves é gerado em seu AWS CloudHSM cluster e a chave privada do par é usada para criar a CSR.

4. Envie a CSR à sua CA. Se estiver usando uma CA do Windows Server, siga estas etapas:

- a. Digite o seguinte comando para abrir a ferramenta CA:

```
certsrv.msc
```

- b. Na nova janela, clique com o botão direito do mouse no nome do servidor da CA. Escolha All Tasks (Todas as tarefas) e depois escolha Submit new request (Enviar nova solicitação).
- c. Navegue até o local do `request.csr` e escolha Open (Abrir).
- d. Navegue até a pasta Pending Requests (Solicitações pendentes) expandindo o menu Server CA (CA do servidor). Clique com o botão direito do mouse na solicitação que você acabou de criar e, em All Tasks (Todas as tarefas), escolha Issue (Problema).
- e. Agora, navegue até a pasta Issued Certificates (Certificados emitidos) (acima da pasta Pending Requests (Solicitações pendentes)).
- f. Escolha Open (Abrir) para visualizar o certificado e depois escolha a guia Details (Detalhes).

- g. Escolha Copy to File (Copiar no arquivo) para iniciar o Certificate Export Wizard (Assistente de exportação da CA). Salve o arquivo X.509 codificado por DER em um local seguro como `signedCertificate.cer`.
- h. Saia da ferramenta CA e use o comando a seguir, que move o arquivo do certificado para o armazenamento de certificados pessoais no Windows. Ele pode então ser usado por outras aplicações.

```
certreq.exe -accept signedCertificate.cer
```

Agora, você pode usar o certificado importado para [Assinar um arquivo](#).

Etapa 3: assinar um arquivo

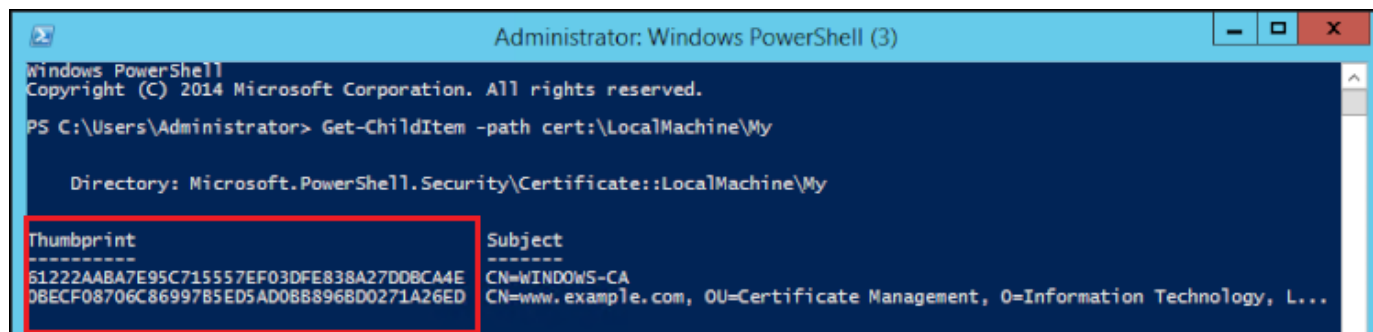
Agora você está pronto para usar SignTool seu certificado importado para assinar seu arquivo de exemplo. Para fazer isso, você precisa saber o hash SHA-1 do certificado ou a impressão digital. A impressão digital é usada para garantir que use SignTool apenas certificados verificados por AWS CloudHSM. Neste exemplo, usamos PowerShell para obter o hash do certificado. Você também pode usar a GUI da CA ou o executável `certutil` do SDK do Windows.

Para obter uma impressão digital do certificado e usá-lo para assinar um arquivo

1. Abra PowerShell como administrador e execute o seguinte comando:

```
Get-ChildItem -path cert:\LocalMachine\My
```

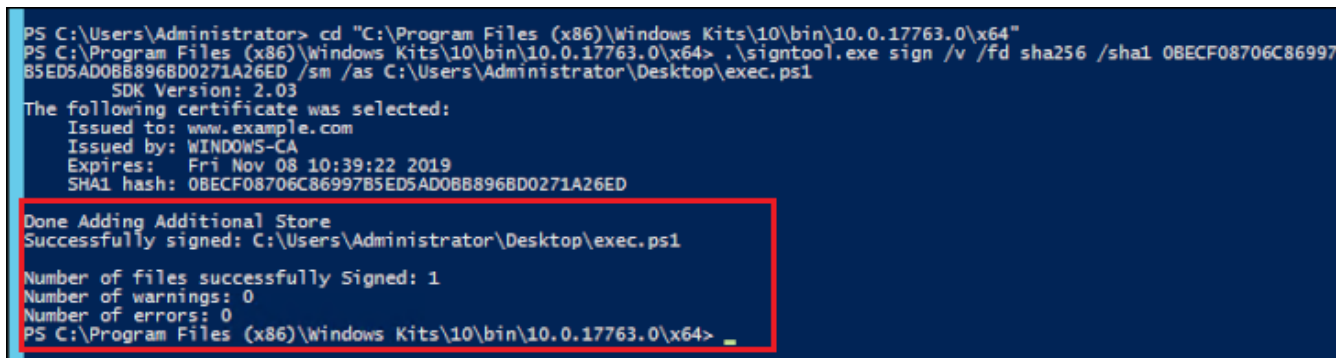
Copie a Thumbprint que é retornada.



2. Navegue até o diretório PowerShell que contém `SignTool.exe`. O local padrão é `C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64`.

3. Finalmente, assine seu arquivo executando o comando a seguir. Se o comando for bem-sucedido, PowerShell retornará uma mensagem de sucesso.

```
signtool.exe sign /v /fd sha256 /sha1 <thumbprint> /sm C:\Users\Administrator\
\Desktop\<test>.ps1
```



```
PS C:\Users\Administrator> cd "C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64"
PS C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64> .\signtool.exe sign /v /fd sha256 /sha1 0BECF08706C86997
85ED5AD0BB8968D0271A26ED /sm /as C:\Users\Administrator\Desktop\exec.ps1
    SDK Version: 2.03
The following certificate was selected:
  Issued to: www.example.com
  Issued by: WINDOWS-CA
  Expires:  Fri Nov 08 10:39:22 2019
  SHA1 hash: 0BECF08706C8699785ED5AD0BB8968D0271A26ED
Done Adding Additional Store
Successfully signed: C:\Users\Administrator\Desktop\exec.ps1
Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
PS C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64> _
```

4. (Opcional) Para verificar a assinatura no arquivo, use o seguinte comando:

```
signtool.exe verify /v /pa C:\Users\Administrator\Desktop\<test>.ps1
```

Integração do Java Keytool e do Jarsigner com AWS CloudHSM

AWS CloudHSM oferece integração com os utilitários Java Keytool e Jarsigner por meio do Client SDK 3 e do Client SDK 5. As etapas para usar essas ferramentas variam de acordo com a versão do Client SDK na qual você baixou atualmente. As seções a seguir fornecem informações sobre cada SDK.

Tópicos

- [Use o Client SDK 5 para fazer a integração AWS CloudHSM com o Java Keytool e o Jarsigner](#)
- [Use o Client SDK 3 para fazer a integração AWS CloudHSM com o Java Keytool e o Jarsigner](#)

Use o Client SDK 5 para fazer a integração AWS CloudHSM com o Java Keytool e o Jarsigner

AWS CloudHSM O armazenamento de chaves é um armazenamento de chaves JCE para fins especiais que utiliza certificados associados às chaves em seu módulo de segurança de hardware (HSM) por meio de ferramentas de terceiros, como e. keytool jarsigner AWS CloudHSM não

armazena certificados no HSM, pois os certificados são dados públicos e não confidenciais. O armazenamento de AWS CloudHSM chaves armazena os certificados em um arquivo local e mapeia os certificados para as chaves correspondentes no seu HSM.

Quando você usa o armazenamento de AWS CloudHSM chaves para gerar novas chaves, nenhuma entrada é gerada no arquivo de armazenamento de chaves local — as chaves são criadas no HSM. Da mesma forma, quando você usa o repositório de chaves do AWS CloudHSM para procurar chaves, a pesquisa é transmitida para o HSM. Quando você armazena certificados no armazenamento de AWS CloudHSM chaves, o provedor verifica se existe um par de chaves com o alias correspondente no HSM e, em seguida, associa o certificado fornecido ao par de chaves correspondente.

Tópicos

- [Pré-requisitos para integração AWS CloudHSM com Java Keytool e Jarsigner usando o Client SDK 5](#)
- [Use o armazenamento de AWS CloudHSM chaves com o keytool usando o Client SDK 5](#)
- [Use o armazenamento de AWS CloudHSM chaves com o Jarsigner usando o Client SDK 5](#)
- [Problemas conhecidos para AWS CloudHSM integração Java Keytool e Jarsigner usando o Client SDK 5](#)

Pré-requisitos para integração AWS CloudHSM com Java Keytool e Jarsigner usando o Client SDK 5

Para usar o armazenamento de AWS CloudHSM chaves, primeiro você deve inicializar e configurar o SDK do AWS CloudHSM JCE. Para isso, siga as etapas a seguir.

Etapas 1: Instalar o JCE

Para instalar o JCE, incluindo os pré-requisitos AWS CloudHSM do cliente, siga as etapas para [instalar](#) a biblioteca Java.

Etapas 2: Adicionar credenciais de login do HSM a variáveis de ambiente

Configure variáveis de ambiente para conter suas credenciais de login do HSM.

Linux

```
$ export HSM_USER=<HSM user name>
```

```
$ export HSM_PASSWORD=<HSM password>
```

Windows

```
PS C:\> $Env:HSM_USER=<HSM user name>
```

```
PS C:\> $Env:HSM_PASSWORD=<HSM password>
```

Note

O AWS CloudHSM JCE oferece várias opções de login. Para usar o armazenamento de AWS CloudHSM chaves com aplicativos de terceiros, você deve usar o login implícito com variáveis de ambiente. Se você quiser usar o login explícito por meio do código do aplicativo, deverá criar seu próprio aplicativo usando o armazenamento de AWS CloudHSM chaves. Para obter informações adicionais, consulte o artigo sobre [Como usar o armazenamento de AWS CloudHSM chaves](#).

Etapa 3: Registrar o provedor JCE

Para registrar o provedor JCE na CloudProvider configuração Java, siga estas etapas:

1. Abra o arquivo de configuração `java.security` em sua instalação Java para edição.
2. No arquivo de configuração `java.security`, adicione `com.amazonaws.cloudhsm.jce.provider.CloudHsmProvider` como o último provedor. Por exemplo, se houver nove provedores no arquivo `java.security`, adicione o provedor a seguir como o último provedor na seção:

```
security.provider.10=com.amazonaws.cloudhsm.jce.provider.CloudHsmProvider
```

Note

Adicionar o AWS CloudHSM provedor como uma prioridade mais alta pode afetar negativamente o desempenho do seu sistema, pois o AWS CloudHSM provedor será priorizado para operações que possam ser transferidas com segurança para o software.

Como prática recomendada, sempre especifique o provedor que você deseja usar para uma operação, seja ele AWS CloudHSM ou um provedor baseado em software.

 Note

A especificação de opções de linha de comando `-providerName`, `-providerclass` e `-providerpath` ao gerar chaves usando `keytool` com o armazenamento de chave do AWS CloudHSM pode causar erros.

Use o armazenamento de AWS CloudHSM chaves com o `keytool` usando o Client SDK 5

[Keytool](#) é um utilitário de linha de comando popular para tarefas comuns de chave e certificado. Um tutorial completo sobre o `keytool` está fora do escopo da documentação do AWS CloudHSM. Este artigo explica os parâmetros específicos que você deve usar com várias funções de ferramentas-chave ao utilizá-las AWS CloudHSM como raiz de confiança por meio do armazenamento de AWS CloudHSM chaves.

Ao usar o `keytool` com o armazenamento de AWS CloudHSM chaves, especifique os seguintes argumentos para qualquer comando `keytool`:

Linux

```
-storetype CLLOUDHSM -J-classpath< '-J/opt/cloudhsm/java/*'>
```

Windows

```
-storetype CLLOUDHSM -J-classpath<'-J"C:\Program Files\Amazon\CloudHSM\java\*"'">
```

Se você quiser criar um novo arquivo de armazenamento de chaves usando o armazenamento de AWS CloudHSM chaves, consulte [Use o AWS CloudHSM KeyStore for AWS CloudHSM Client SDK 3](#). Para usar um armazenamento de chaves existente, especifique seu nome (incluindo o caminho) usando o argumento `keystore` para `keytool`. Se você especificar um arquivo de armazenamento de chaves inexistente em um comando `keytool`, o armazenamento de AWS CloudHSM chaves criará um novo arquivo de armazenamento de chaves.

Crie novas AWS CloudHSM chaves com o keytool

Você pode usar o keytool para gerar RSA, AES e o DESede tipo de chave compatível com o SDK do AWS CloudHSM JCE.

Important

Uma chave gerada por meio do keytool é gerada no software e depois importada AWS CloudHSM como uma chave persistente e extraível.

É altamente recomendável gerar chaves não exportáveis fora do keytool e importar certificados correspondentes para o repositório de chaves. Se você usar chaves RSA ou EC extraíveis por meio do keytool e do Jarsigner, os provedores exportam as chaves do AWS CloudHSM e, em seguida, usam a chave localmente para operações de assinatura.

Se você tiver várias instâncias de cliente conectadas ao seu AWS CloudHSM cluster, saiba que importar um certificado no armazenamento de chaves de uma instância cliente não disponibilizará automaticamente os certificados em outras instâncias de cliente. Para registrar a chave e os certificados associados em cada instância do cliente, você precisa executar um aplicativo Java conforme descrito em [the section called “Gerar um CSR usando keytool”](#). Como alternativa, você pode fazer as alterações necessárias em um cliente e copiar o arquivo repositório de chaves resultante para todas as outras instâncias de cliente.

Exemplo 1: para gerar uma chave AES-256 simétrica e salvá-la em um arquivo armazenamento de chave chamado “example_keystore.store”, no diretório de trabalho. *<secret label>* Substitua por uma etiqueta exclusiva.

Linux

```
$ keytool -genseckey -alias <secret label> -keyalg aes \
-keysize 256 -keystore example_keystore.store \
-storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
```

Windows

```
PS C:\> keytool -genseckey -alias <secret label> -keyalg aes `
-keysize 256 -keystore example_keystore.store `
-storetype CloudHSM -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Exemplo 2: para gerar um par de chaves RSA 2048 e salvá-lo em um arquivo armazenamento de chave chamado “example_keystore.store”, no diretório de trabalho. *<RSA key pair label>* Substitua por uma etiqueta exclusiva.

Linux

```
$ keytool -genkeypair -alias <RSA key pair label> \  
-keyalg rsa -keysize 2048 \  
-sigalg sha512withrsa \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -genkeypair -alias <RSA key pair label> \  
-keyalg rsa -keysize 2048 \  
-sigalg sha512withrsa \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Você pode encontrar uma lista de [algoritmos de assinatura suportados](#) na biblioteca Java.

Exclua uma AWS CloudHSM chave usando o keytool

O armazenamento de AWS CloudHSM chaves não suporta a exclusão de chaves. Você pode excluir chaves usando o método de destruição da [interface Destrutível](#).

```
((Destroyable) key).destroy();
```

Gere uma AWS CloudHSM CSR usando o keytool

Você recebe a maior flexibilidade na geração de um pedido de assinatura de certificado (CSR) se você usar o [Mecanismo AWS CloudHSM dinâmico OpenSSL para cliente SDK 5](#). O comando a seguir usa o keytool para gerar um CSR para um par de chaves com o alias example-key-pair.

Linux

```
$ keytool -certreq -alias <key pair label> \  
-file my_csr.csr \  

```

```
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -certreq -alias <key pair label> `\  
-file my_csr.csr `\  
-keystore example_keystore.store `\  
-storetype CLOUDHSM `\  
-J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Note

Para usar um par de chaves da keytool, esse par de chaves deve ter uma entrada no arquivo de repositório de chaves especificado. Se você quiser usar um par de chaves que foi gerado fora do keytool, você deve importar os metadados de chave e certificado para o repositório de chaves. Para obter instruções sobre como importar os dados do armazenamento de chave, consulte [the section called “Usar o keytool para importar certificados para o repositório de chaves”](#)

Use o keytool para importar certificados intermediários e raiz para o armazenamento de AWS CloudHSM chaves

Para importar um certificado de CA AWS CloudHSM, você deve habilitar a verificação de uma cadeia de certificados completa em um certificado recém-importado. O seguinte comando mostra um exemplo.

Linux

```
$ keytool -import -trustcacerts -alias rootCAcert \  
-file rootCAcert.cert -keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -import -trustcacerts -alias rootCAcert `
```

```
-file rootCAcert.cert -keystore example_keystore.store `
-storetype CLOUDHSM `
-J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Se você conectar várias instâncias do cliente ao seu AWS CloudHSM cluster, importar um certificado no armazenamento de chaves de uma instância do cliente não disponibilizará automaticamente o certificado em outras instâncias do cliente. Você deve importar o certificado em cada instância do cliente.

Use o keytool para excluir certificados do armazenamento de AWS CloudHSM chaves

O comando a seguir mostra um exemplo de como excluir um AWS CloudHSM certificado de um armazenamento de chaves Java keytool.

Linux

```
$ keytool -delete -alias mydomain \
-keystore example_keystore.store \
-storetype CLOUDHSM \
-J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -delete -alias mydomain `
-keystore example_keystore.store `
-storetype CLOUDHSM `
-J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Se você conectar várias instâncias do cliente ao seu AWS CloudHSM cluster, a exclusão de um certificado no armazenamento de chaves de uma instância do cliente não removerá automaticamente o certificado de outras instâncias do cliente. Você deve excluir o certificado em cada instância de cliente.

Importe um certificado funcional para o armazenamento de AWS CloudHSM chaves usando o keytool

Depois que uma solicitação de assinatura de certificado (CSR) for assinada, você poderá importá-la para o repositório de chaves do AWS CloudHSM e associá-la ao par de chaves apropriado. O comando a seguir fornece um exemplo.

Linux

```
$ keytool -importcert -noprompt -alias <key pair label> \  
-file my_certificate.crt \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -importcert -noprompt -alias <key pair label> \  
-file my_certificate.crt \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

O alias deve ser um par de chaves com um certificado associado no repositório de chaves. Se a chave for gerada fora da ferramenta de chave ou for gerada em uma instância de cliente diferente, você deve primeiro importar os metadados de chave e certificado para o repositório de chaves.

A cadeia de certificados deve ser verificável. Se você não conseguir verificar o certificado, talvez seja necessário importar o certificado de assinatura (autoridade de certificação) para o repositório de chaves para que a cadeia possa ser verificada.

Exportar um certificado AWS CloudHSM usando o keytool

O exemplo a seguir gera um certificado no formato binário X.509. Para exportar um certificado legível por humanos de AWS CloudHSM, adicione `-rfc` ao `-exportcert` comando.

Linux

```
$ keytool -exportcert -alias <key pair label> \  
-file my_exported_certificate.crt \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -exportcert -alias <key pair label> `
```



```
-file my_exported_certificate.crt `
-keystore example_keystore.store `
-storetype CLOUDHSM `
-J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Use o armazenamento de AWS CloudHSM chaves com o Jarsigner usando o Client SDK 5

Jarsigner é um utilitário de linha de comando popular para assinar arquivos JAR usando uma chave armazenada de forma segura em um HSM. Um tutorial completo sobre o Jarsigner está fora do escopo da AWS CloudHSM documentação. Esta seção explica os parâmetros do Jarsigner que você deve usar para assinar e verificar assinaturas AWS CloudHSM como raiz de confiança por meio do AWS CloudHSM armazenamento de chaves.

Configure AWS CloudHSM chaves e certificados com o Jarsigner

Antes de assinar arquivos AWS CloudHSM JAR com o Jarsigner, certifique-se de ter configurado ou concluído as seguintes etapas:

1. Siga as orientações nos [AWS CloudHSM pré-requisitos de repositório de chaves](#).
2. Configure suas chaves de assinatura e os certificados associados e a cadeia de certificados, que devem ser armazenados no armazenamento de AWS CloudHSM chaves da instância atual do servidor ou cliente. Crie as chaves no AWS CloudHSM e, em seguida, importe os metadados associados para o seu armazenamento de AWS CloudHSM chaves. Se você quiser usar o keytool para configurar as chaves e certificados, consulte [the section called “Criar novas chaves do com keytool”](#). Se você usar várias instâncias de cliente para assinar sua JARs, crie a chave e importe a cadeia de certificados. Em seguida, copie o arquivo de repositório de chaves resultante para cada instância do cliente. Se você gerar novas chaves com frequência, talvez seja mais fácil importar certificados individualmente para cada instância do cliente.
3. Toda a cadeia de certificados deve ser verificável. Para que a cadeia de certificados seja verificável, talvez seja necessário adicionar o certificado CA e os certificados intermediários ao armazenamento de AWS CloudHSM chaves. Consulte o trecho de código em [the section called “Assinar um arquivo JAR”](#) para obter instruções sobre como usar o código Java para verificar a cadeia de certificados. Se preferir, você pode usar o keytool para importar certificados. Para obter instruções sobre como usar o keytool, consulte [the section called “Usar o keytool para importar certificados para o repositório de chaves”](#).

Assine um arquivo JAR usando AWS CloudHSM e Jarsigner

Use o comando a seguir para assinar um arquivo JAR usando o AWS CloudHSM Jarsigner:

Linux;

Para OpenJDK 8

```
jarsigner -keystore example_keystore.store \  
-signedjar signthisclass_signed.jar \  
-sigalg sha512withrsa \  
-storetype CloudHSM \  
-J-classpath '-J/opt/cloudhsm/java/*:/usr/lib/jvm/java-1.8.0/lib/tools.jar' \  
-J-Djava.library.path=/opt/cloudhsm/lib \  
signthisclass.jar <key pair label>
```

Para OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -keystore example_keystore.store \  
-signedjar signthisclass_signed.jar \  
-sigalg sha512withrsa \  
-storetype CloudHSM \  
-J-classpath '-J/opt/cloudhsm/java/*' \  
-J-Djava.library.path=/opt/cloudhsm/lib \  
signthisclass.jar <key pair label>
```

Windows

Para Open JDK8

```
jarsigner -keystore example_keystore.store \  
-signedjar signthisclass_signed.jar \  
-sigalg sha512withrsa \  
-storetype CloudHSM \  
-J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*;C:\Program Files\Java\  
\jdk1.8.0_331\lib\tools.jar' \  
"-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" \  
signthisclass.jar <key pair label>
```

Para OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -keystore example_keystore.store `
-signedjar signthisclass_signed.jar `
-sigalg sha512withrsa `
-storetype CloudHSM `
-J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*' `
"-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
signthisclass.jar <key pair label>
```

Use o seguinte comando para verificar um JAR assinado:

Linux

Para Open JDK8

```
jarsigner -verify \
-keystore example_keystore.store \
-sigalg sha512withrsa \
-storetype CloudHSM \
-J-classpath '-J/opt/cloudhsm/java/*:/usr/lib/jvm/java-1.8.0/lib/tools.jar' \
-J-Djava.library.path=/opt/cloudhsm/lib \
signthisclass_signed.jar <key pair label>
```

Para OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -verify \
-keystore example_keystore.store \
-sigalg sha512withrsa \
-storetype CloudHSM \
-J-classpath '-J/opt/cloudhsm/java/*' \
-J-Djava.library.path=/opt/cloudhsm/lib \
signthisclass_signed.jar <key pair label>
```

Windows

Para OpenJDK 8

```
jarsigner -verify `
-keystore example_keystore.store `
```

```
-sigalg sha512withrsa `
-storetype CloudHSM `
-J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*;C:\Program Files\Java
\jdk1.8.0_331\lib\tools.jar' `
"-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
signthisclass_signed.jar <key pair label>
```

Para OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -verify `
-keystore example_keystore.store `
-sigalg sha512withrsa `
-storetype CloudHSM `
-J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*`
"-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
signthisclass_signed.jar <key pair label>
```

Problemas conhecidos para AWS CloudHSM integração Java Keytool e Jarsigner usando o Client SDK 5

A lista a seguir fornece a lista atual de problemas conhecidos de integrações com AWS CloudHSM , Java Keytool e Jarsigner usando o Client SDK 5.

1. Não oferecemos suporte a chaves EC com Keytool e Jarsigner.

Use o Client SDK 3 para fazer a integração AWS CloudHSM com o Java Keytool e o Jarsigner

AWS CloudHSM O armazenamento de chaves é um armazenamento de chaves JCE para fins especiais que utiliza certificados associados às chaves em seu módulo de segurança de hardware (HSM) por meio de ferramentas de terceiros, como e. keytool jarsigner AWS CloudHSM não armazena certificados no HSM, pois os certificados são dados públicos e não confidenciais. O armazenamento de AWS CloudHSM chaves armazena os certificados em um arquivo local e mapeia os certificados para as chaves correspondentes no seu HSM.

Quando você usa o armazenamento de AWS CloudHSM chaves para gerar novas chaves, nenhuma entrada é gerada no arquivo de armazenamento de chaves local — as chaves são criadas no

HSM. Da mesma forma, quando você usa o repositório de chaves do AWS CloudHSM para procurar chaves, a pesquisa é transmitida para o HSM. Quando você armazena certificados no armazenamento de AWS CloudHSM chaves, o provedor verifica se existe um par de chaves com o alias correspondente no HSM e, em seguida, associa o certificado fornecido ao par de chaves correspondente.

Tópicos

- [Pré-requisitos para integração AWS CloudHSM com Java Keytool e Jarsigner usando o Client SDK 3](#)
- [Use o armazenamento de AWS CloudHSM chaves com o keytool usando o Client SDK 3](#)
- [Use o armazenamento de AWS CloudHSM chaves com o Jarsigner usando o Client SDK 3](#)
- [Problemas conhecidos para AWS CloudHSM integração Java Keytool e Jarsigner usando o Client SDK 3](#)
- [Registre chaves pré-existentes com o armazenamento de AWS CloudHSM chaves](#)

Pré-requisitos para integração AWS CloudHSM com Java Keytool e Jarsigner usando o Client SDK 3

Para usar o armazenamento de AWS CloudHSM chaves, primeiro você deve inicializar e configurar o SDK do AWS CloudHSM JCE. Para isso, siga as etapas a seguir.

Etapa 1: Instalar o JCE

Para instalar o JCE, incluindo os pré-requisitos AWS CloudHSM do cliente, siga as etapas para [instalar](#) a biblioteca Java.

Etapa 2: Adicionar credenciais de login do HSM a variáveis de ambiente

Configure variáveis de ambiente para conter suas credenciais de login do HSM.

```
export HSM_PARTITION=PARTITION_1
export HSM_USER=<HSM user name>
export HSM_PASSWORD=<HSM password>
```

 Note

O CloudHSM JCE oferece várias opções de login. Para usar o armazenamento de AWS CloudHSM chaves com aplicativos de terceiros, você deve usar o login implícito com variáveis de ambiente. Se você quiser usar o login explícito por meio do código do aplicativo, deverá criar seu próprio aplicativo usando o armazenamento de AWS CloudHSM chaves. Para obter informações adicionais, consulte o artigo sobre [Como usar o armazenamento de AWS CloudHSM chaves](#).

Etapa 3: Registrar o provedor JCE

Para registrar o provedor JCE, na CloudProvider configuração Java.

1. Abra o arquivo de configuração `java.security` em sua instalação Java para edição.
2. No arquivo de configuração `java.security`, adicione `com.cavium.provider.CaviumProvider` como o último provedor. Por exemplo, se houver nove provedores no arquivo `java.security`, adicione o provedor a seguir como o último provedor na seção. Adicionar o provedor Cavium como prioridade maior pode afetar negativamente o desempenho do seu sistema.

```
security.provider.10=com.cavium.provider.CaviumProvider
```

 Note

Usuários avançados podem estar acostumados a especificar opções de linha de comando `-providerName`, `-providerclass` e `-providerpath` ao usar o `keytool`, em vez de atualizar o arquivo de configuração de segurança. Se você tentar especificar as opções da linha de comando ao gerar chaves com o armazenamento de AWS CloudHSM chaves, isso causará erros.

Use o armazenamento de AWS CloudHSM chaves com o `keytool` usando o Client SDK 3

[Keytool](#) é um utilitário de linha de comando popular para tarefas comuns de chave e certificado em sistemas Linux. Um tutorial completo sobre o `keytool` está fora do escopo da documentação do AWS CloudHSM . Este artigo explica os parâmetros específicos que você deve usar com várias

funções de ferramentas-chave ao utilizá-las AWS CloudHSM como raiz de confiança por meio do armazenamento de AWS CloudHSM chaves.

Ao usar o keytool com o armazenamento de AWS CloudHSM chaves, especifique os seguintes argumentos para qualquer comando keytool:

```
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*' \  
-J-Djava.library.path=/opt/cloudhsm/lib
```

Se você quiser criar um novo arquivo de armazenamento de chaves usando o armazenamento de AWS CloudHSM chaves, consulte [Use o AWS CloudHSM KeyStore for AWS CloudHSM Client SDK 3](#). Para usar um armazenamento de chaves existente, especifique seu nome (incluindo o caminho) usando o argumento keystore para keytool. Se você especificar um arquivo de armazenamento de chaves inexistente em um comando keytool, o armazenamento de AWS CloudHSM chaves criará um novo arquivo de armazenamento de chaves.

Crie novas AWS CloudHSM chaves com o keytool

Você pode usar o keytool para gerar qualquer tipo de chave compatível com o SDK do AWS CloudHSM JCE. Veja uma lista completa de chaves e comprimentos no artigo [Chaves Suportadas](#) na Biblioteca Java.

 Important

Uma chave gerada por meio do keytool é gerada no software e depois importada AWS CloudHSM como uma chave persistente e extraível.

As instruções para criar chaves não extraíveis diretamente no módulo de segurança de hardware (HSM) e depois usá-las com o keytool ou o Jarsigner são mostradas na amostra de código em [Registrando](#) chaves pré-existentes no Key Store. AWS CloudHSM É altamente recomendável gerar chaves não exportáveis fora do keytool e importar certificados correspondentes para o repositório de chaves. Se você usar chaves RSA ou EC extraíveis por meio de keytool e jarsigner, os provedores exportarão as chaves do e, em seguida, usarão a chave AWS CloudHSM localmente para operações de assinatura.

Se você tiver várias instâncias de cliente conectadas ao cluster do CloudHSM, esteja ciente de que importar um certificado no repositório de chaves de uma instância de cliente não disponibilizará

automaticamente os certificados em outras instâncias de cliente. Para registrar a chave e os certificados associados em cada instância do cliente, você precisa executar um aplicativo Java conforme descrito em [Gerar um CSR usando Keytool](#). Como alternativa, você pode fazer as alterações necessárias em um cliente e copiar o arquivo repositório de chaves resultante para todas as outras instâncias de cliente.

Exemplo 1: para gerar uma chave AES-256 simétrica e salvá-la em um arquivo armazenamento de chave chamado “example_keystore.store”, no diretório de trabalho. *<secret label>* Substitua por uma etiqueta exclusiva.

```
keytool -genseckey -alias <secret label> -keyalg aes \
-keysize 256 -keystore example_keystore.store \
-storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
-J-Djava.library.path=/opt/cloudhsm/lib/
```

Exemplo 2: para gerar um par de chaves RSA 2048 e salvá-lo em um arquivo armazenamento de chave chamado “example_keystore.store”, no diretório de trabalho. *<RSA key pair label>* Substitua por uma etiqueta exclusiva.

```
keytool -genkeypair -alias <RSA key pair label> \
-keyalg rsa -keysize 2048 \
-sigalg sha512withrsa \
-keystore example_keystore.store \
-storetype CLOUDHSM \
-J-classpath '-J/opt/cloudhsm/java/*' \
-J-Djava.library.path=/opt/cloudhsm/lib/
```

Exemplo 3: para gerar um par de chaves p256 ED e salvá-lo em um arquivo armazenamento de chave chamado “example_keystore.store”, no diretório de trabalho. *<ec key pair label>* Substitua por uma etiqueta exclusiva.

```
keytool -genkeypair -alias <ec key pair label> \
-keyalg ec -keysize 256 \
-sigalg SHA512withECDSA \
-keystore example_keystore.store \
-storetype CLOUDHSM \
-J-classpath '-J/opt/cloudhsm/java/*' \
-J-Djava.library.path=/opt/cloudhsm/lib/
```

Você pode encontrar uma lista de [algoritmos de assinatura suportados](#) na biblioteca Java.

Exclua uma AWS CloudHSM chave usando o keytool

O armazenamento de AWS CloudHSM chaves não suporta a exclusão de chaves. Para excluir a chave, você deve usar a `deleteKey` função AWS CloudHSM da ferramenta de linha de comando, [Excluir uma AWS CloudHSM chave usando KMU](#).

Gere uma AWS CloudHSM CSR usando o keytool

Você recebe a maior flexibilidade na geração de um pedido de assinatura de certificado (CSR) se você usar o [Mecanismo AWS CloudHSM dinâmico OpenSSL para cliente SDK 5](#). O comando a seguir usa o keytool para gerar um CSR para um par de chaves com o alias `example-key-pair`.

```
keytool -certreq -alias <key pair label> \
        -file example_csr.csr \
        -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Note

Para usar um par de chaves da keytool, esse par de chaves deve ter uma entrada no arquivo de repositório de chaves especificado. Se você quiser usar um par de chaves que foi gerado fora do keytool, você deve importar os metadados de chave e certificado para o repositório de chaves. Para obter instruções sobre como importar os dados do armazenamento de chaves, consulte [Importação de certificados intermediários e raiz para o AWS CloudHSM Key Store](#) usando o Keytool.

Use o keytool para importar certificados intermediários e raiz para o armazenamento de AWS CloudHSM chaves

Para importar um certificado CA para AWS CloudHSM, você deve habilitar a verificação de uma cadeia de certificados completa em um certificado recém-importado. O seguinte comando mostra um exemplo.

```
keytool -import -trustcacerts -alias rootCAcert \
        -file rootCAcert.cert -keystore example_keystore.store \
        -storetype CLOUDHSM \
```

```
-J-classpath '-J/opt/cloudhsm/java/*' \  
-J-Djava.library.path=/opt/cloudhsm/lib/
```

Se você conectar várias instâncias do cliente ao seu AWS CloudHSM cluster, importar um certificado no armazenamento de chaves de uma instância do cliente não disponibilizará automaticamente o certificado em outras instâncias do cliente. Você deve importar o certificado em cada instância do cliente.

Use o keytool para excluir certificados do armazenamento de AWS CloudHSM chaves

O comando a seguir mostra um exemplo de como excluir um AWS CloudHSM certificado de um armazenamento de chaves Java keytool.

```
keytool -delete -alias mydomain -keystore \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*' \  
-J-Djava.library.path=/opt/cloudhsm/lib/
```

Se você conectar várias instâncias do cliente ao seu AWS CloudHSM cluster, a exclusão de um certificado no armazenamento de chaves de uma instância do cliente não removerá automaticamente o certificado de outras instâncias do cliente. Você deve excluir o certificado em cada instância de cliente.

Importe um certificado funcional para o armazenamento de AWS CloudHSM chaves usando o keytool

Depois que uma solicitação de assinatura de certificado (CSR) for assinada, você poderá importá-la para o repositório de chaves do AWS CloudHSM e associá-la ao par de chaves apropriado. O comando a seguir fornece um exemplo.

```
keytool -importcert -noprompt -alias <key pair label> \  
-file example_certificate.crt \  
-keystore example_keystore.store \  
-storetype CLOUDHSM \  
-J-classpath '-J/opt/cloudhsm/java/*' \  
-J-Djava.library.path=/opt/cloudhsm/lib/
```

O alias deve ser um par de chaves com um certificado associado no repositório de chaves. Se a chave for gerada fora da ferramenta de chave ou for gerada em uma instância de cliente diferente,

you must first import the key and certificate metadata to the key store. For instructions on how to import the certificate metadata, consult the code example in [Registering pre-existing keys](#) with the Key Store. AWS CloudHSM

The certificate chain must be verifiable. If you cannot verify the certificate, you may need to import the signature certificate (certifying authority) to the key store so that the chain can be verified.

Export an AWS CloudHSM certificate using the keytool

The example below generates a certificate in the X.509 binary format. To export a human-readable certificate from AWS CloudHSM, add `-rfc` to the `-exportcert` command.

```
keytool -exportcert -alias <key pair label> \
  -file example_exported_certificate.crt \
  -keystore example_keystore.store \
  -storetype CLOUDHSM \
  -J-classpath '-J/opt/cloudhsm/java/*' \
  -J-Djava.library.path=/opt/cloudhsm/lib/
```

Use the AWS CloudHSM key storage with Jarsigner using the Client SDK 3

Jarsigner is a popular command-line utility for signing JAR files using a key stored securely in an HSM. A complete tutorial on Jarsigner is out of scope for the AWS CloudHSM documentation. This section explains the Jarsigner parameters that you must use to sign and verify AWS CloudHSM signatures as the root of trust by using the AWS CloudHSM key storage.

Configure AWS CloudHSM keys and certificates with Jarsigner

Before signing AWS CloudHSM JAR files with Jarsigner, be sure to configure or complete the following steps:

1. Follow the instructions in [AWS CloudHSM key store prerequisites](#).
2. Configure your signature keys and associated certificates and the certificate chain, which must be stored in the AWS CloudHSM key storage of the current server or client. Create the keys in AWS CloudHSM and, subsequently, import the associated metadata to your AWS CloudHSM key storage. Use the example

de código em [Registrando chaves pré-existentes com o AWS CloudHSM Key Store](#) para importar metadados para o armazenamento de chaves. Se você quiser usar o keytool para configurar as chaves e certificados, consulte [Crie novas AWS CloudHSM chaves com o keytool](#). Se você usar várias instâncias de cliente para assinar sua JARs, crie a chave e importe a cadeia de certificados. Em seguida, copie o arquivo de repositório de chaves resultante para cada instância do cliente. Se você gerar novas chaves com frequência, talvez seja mais fácil importar certificados individualmente para cada instância do cliente.

3. Toda a cadeia de certificados deve ser verificável. Para que a cadeia de certificados seja verificável, talvez seja necessário adicionar o certificado CA e os certificados intermediários ao armazenamento de AWS CloudHSM chaves. Consulte o trecho de código em [Assine um arquivo JAR usando AWS CloudHSM e Jarsigner](#) para obter instruções sobre como usar o código Java para verificar a cadeia de certificados. Se preferir, você pode usar o keytool para importar certificados. Para obter instruções sobre como usar o keytool, consulte [Usando o Keytool para importar certificados intermediários e raiz para o AWS CloudHSM Key Store](#).

Assine um arquivo JAR usando AWS CloudHSM e Jarsigner

Use o comando a seguir para assinar um arquivo JAR usando o AWS CloudHSM jarsigner:

```
jarsigner -keystore example_keystore.store \  
    -signedjar signthisclass_signed.jar \  
    -sigalg sha512withrsa \  
    -storetype CloudHSM \  
    -J-classpath '-J/opt/cloudhsm/java/*:/usr/lib/jvm/java-1.8.0/lib/tools.jar' \  
    -J-Djava.library.path=/opt/cloudhsm/lib \  
    signthisclass.jar <key pair label>
```

Use o seguinte comando para verificar um JAR assinado:

```
jarsigner -verify \  
    -keystore example_keystore.store \  
    -sigalg sha512withrsa \  
    -storetype CloudHSM \  
    -J-classpath '-J/opt/cloudhsm/java/*:/usr/lib/jvm/java-1.8.0/lib/tools.jar' \  
    -J-Djava.library.path=/opt/cloudhsm/lib \  
    signthisclass_signed.jar <key pair label>
```

Problemas conhecidos para AWS CloudHSM integração Java Keytool e Jarsigner usando o Client SDK 3

A lista a seguir fornece a lista atual de problemas conhecidos de integrações com Java Keytool AWS CloudHSM e Jarsigner usando o Client SDK 3.

- Ao gerar chaves usando o keytool, o primeiro provedor na configuração do provedor não pode ser CaviumProvider.
- Ao gerar chaves usando keytool, o primeiro provedor (compatível) no arquivo de configuração de segurança é usado para gerar a chave. Geralmente é um provedor de software. Em seguida, a chave gerada recebe um alias e é importada para o AWS CloudHSM HSM como uma chave persistente (token) durante o processo de adição da chave.
- Ao usar o keytool com o armazenamento de AWS CloudHSM chaves, não especifique -providerName nem -providerpath as opções na linha de comando. -providerclass Especifique essas opções no arquivo do provedor de segurança, conforme descrito nos [pré-requisitos do repositório de chaves](#).
- Ao usar chaves EC não extraíveis por meio do keytool e do Jarsigner, o provedor SunEC precisa removed/disabled estar na lista de provedores no arquivo java.security. Se você usar chaves EC extraíveis por meio do keytool e do Jarsigner, os provedores exportarão bits de chave do AWS CloudHSM HSM e usarão a chave localmente para operações de assinatura. Não recomendamos o uso de chaves exportáveis com keytool ou Jarsigner.

Registre chaves pré-existentes com o armazenamento de AWS CloudHSM chaves

Para máxima segurança e flexibilidade em atributos e rótulos, recomendamos que você gere suas chaves de AWS CloudHSM assinatura usando [key_mgmt_util](#). Você também pode usar um aplicativo Java para gerar a chave no AWS CloudHSM.

A seção a seguir fornece um exemplo de código que demonstra como gerar um novo par de chaves no HSM e registrá-lo usando chaves existentes importadas para o AWS CloudHSM armazenamento de chaves. As chaves importadas estão disponíveis para uso com ferramentas de terceiros, como keytool e Jarsigner.

Para usar uma chave pré-existente, modifique o exemplo de código para procurar uma chave por rótulo em vez de gerar uma nova chave. O código de amostra para pesquisar uma chave por rótulo está disponível na [amostra KeyUtilitiesRunner.java](#) em GitHub.

⚠ Important

Registrar uma chave AWS CloudHSM armazenada em um repositório de chaves local não exporta a chave. Quando a chave é registrada, o repositório de chaves registra o alias (ou rótulo) da chave e correlaciona localmente objetos de certificado de armazenamento com um par de chaves no AWS CloudHSM. Desde que o par de chaves seja criado como não exportável, os bits de chave não sairão do HSM.

```
//
// Copyright 2018 Amazon.com, Inc. or its affiliates. All Rights Reserved.
//
// Permission is hereby granted, free of charge, to any person obtaining a copy of
// this
// software and associated documentation files (the "Software"), to deal in the
// Software
// without restriction, including without limitation the rights to use, copy, modify,
// merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
// permit persons to whom the Software is furnished to do so.
//
// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
// INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
// PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
// HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
// OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
// SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
//

package com.amazonaws.cloudhsm.examples;

import com.cavium.key.CaviumKey;
import com.cavium.key.parameter.CaviumAESKeyGenParameterSpec;
import com.cavium.key.parameter.CaviumRSAKeyGenParameterSpec;
import com.cavium.asn1.Encoder;
import com.cavium.cfm2.Util;

import javax.crypto.KeyGenerator;

import java.io.ByteArrayInputStream;
```

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.FileNotFoundException;

import java.math.BigInteger;

import java.security.*;
import java.security.cert.Certificate;
import java.security.cert.CertificateException;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;
import java.security.interfaces.RSAPrivateKey;
import java.security.interfaces.RSAPublicKey;
import java.security.KeyStore.PasswordProtection;
import java.security.KeyStore.PrivateKeyEntry;
import java.security.KeyStore.Entry;

import java.util.Calendar;
import java.util.Date;
import java.util.Enumuration;

//
// KeyStoreExampleRunner demonstrates how to load a keystore, and associate a
// certificate with a
// key in that keystore.
//
// This example relies on implicit credentials, so you must setup your environment
// correctly.
//
// https://docs.aws.amazon.com/cloudhsm/latest/userguide/java-library-
install.html#java-library-credentials
//

public class KeyStoreExampleRunner {

    private static byte[] COMMON_NAME_OID = new byte[] { (byte) 0x55, (byte) 0x04,
(byte) 0x03 };
    private static byte[] COUNTRY_NAME_OID = new byte[] { (byte) 0x55, (byte) 0x04,
(byte) 0x06 };
    private static byte[] LOCALITY_NAME_OID = new byte[] { (byte) 0x55, (byte) 0x04,
(byte) 0x07 };
    private static byte[] STATE_OR_PROVINCE_NAME_OID = new byte[] { (byte) 0x55,
(byte) 0x04, (byte) 0x08 };
```

```

    private static byte[] ORGANIZATION_NAME_OID = new byte[] { (byte) 0x55, (byte)
0x04, (byte) 0x0A };
    private static byte[] ORGANIZATION_UNIT_OID = new byte[] { (byte) 0x55, (byte)
0x04, (byte) 0x0B };

    private static String helpString = "KeyStoreExampleRunner%n" +
        "This sample demonstrates how to load and store keys using a keystore.%n%n"
+
        "Options%n" +
        "\t--help\t\t\tDisplay this message.%n" +
        "\t--store <filename>\t\tPath of the keystore.%n" +
        "\t--password <password>\t\tPassword for the keystore (not your CU
password).%n" +
        "\t--label <label>\t\tLabel to store the key and certificate under.%n" +
        "\t--list\t\t\tList all the keys in the keystore.%n%n";

    public static void main(String[] args) throws Exception {
        Security.addProvider(new com.cavium.provider.CaviumProvider());
        KeyStore keyStore = KeyStore.getInstance("CloudHSM");

        String keystoreFile = null;
        String password = null;
        String label = null;
        boolean list = false;
        for (int i = 0; i < args.length; i++) {
            String arg = args[i];
            switch (args[i]) {
                case "--store":
                    keystoreFile = args[++i];
                    break;
                case "--password":
                    password = args[++i];
                    break;
                case "--label":
                    label = args[++i];
                    break;
                case "--list":
                    list = true;
                    break;
                case "--help":
                    help();
                    return;
            }
        }
    }
}

```



```

    if (null == keystoreFile || null == password) {
        help();
        return;
    }

    if (list) {
        listKeys(keystoreFile, password);
        return;
    }

    if (null == label) {
        label = "Keystore Example Keypair";
    }

    //
    // This call to keyStore.load() will open the pkcs12 keystore with the supplied
    // password and connect to the HSM. The CU credentials must be specified using
    // standard CloudHSM login methods.
    //
    try {
        FileInputStream instream = new FileInputStream(keystoreFile);
        keyStore.load(instream, password.toCharArray());
    } catch (FileNotFoundException ex) {
        System.err.println("Keystore not found, loading an empty store");
        keyStore.load(null, null);
    }

    PasswordProtection passwd = new PasswordProtection(password.toCharArray());
    System.out.println("Searching for example key and certificate...");

    PrivateKeyEntry keyEntry = (PrivateKeyEntry) keyStore.getEntry(label, passwd);
    if (null == keyEntry) {
        //
        // No entry was found, so we need to create a key pair and associate a
certificate.
        // The private key will get the label passed on the command line. The
keystore alias
        // needs to be the same as the private key label. The public key will have
":public"
        // appended to it. The alias used in the keystore will We associate the
certificate
        // with the private key.
        //

```

```

        System.out.println("No entry found, creating...");
        KeyPair kp = generateRSAKeyPair(2048, label + ":public", label);
        System.out.printf("Created a key pair with the handles %d/%d\n",
            ((CaviumKey) kp.getPrivate()).getHandle(), ((CaviumKey) kp.getPublic()).getHandle());

        //
        // Generate a certificate and associate the chain with the private key.
        //
        Certificate self_signed_cert = generateCert(kp);
        Certificate[] chain = new Certificate[1];
        chain[0] = self_signed_cert;
        PrivateKeyEntry entry = new PrivateKeyEntry(kp.getPrivate(), chain);

        //
        // Set the entry using the label as the alias and save the store.
        // The alias must match the private key label.
        //
        keyStore.setEntry(label, entry, passwd);

        FileOutputStream outstream = new FileOutputStream(keystoreFile);
        keyStore.store(outstream, password.toCharArray());
        outstream.close();

        keyEntry = (PrivateKeyEntry) keyStore.getEntry(label, passwd);
    }

    long handle = ((CaviumKey) keyEntry.getPrivateKey()).getHandle();
    String name = keyEntry.getCertificate().toString();
    System.out.printf("Found private key %d with certificate %s\n", handle, name);
}

private static void help() {
    System.out.println(helpString);
}

//
// Generate a non-extractable / non-persistent RSA keypair.
// This method allows us to specify the public and private labels, which
// will make KeyStore aliases easier to understand.
//
public static KeyPair generateRSAKeyPair(int keySizeInBits, String publicLabel,
String privateLabel)
    throws InvalidAlgorithmParameterException, NoSuchAlgorithmException,
    NoSuchProviderException {

```

```

        boolean isExtractable = false;
        boolean isPersistent = false;
        KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("rsa", "Cavium");
        CaviumRSAKeyGenParameterSpec spec = new
CaviumRSAKeyGenParameterSpec(keySizeInBits, new BigInteger("65537"), publicLabel,
privateLabel, isExtractable, isPersistent);

        keyPairGen.initialize(spec);

        return keyPairGen.generateKeyPair();
    }

    //
    // Generate a certificate signed by a given keypair.
    //
    private static Certificate generateCert(KeyPair kp) throws CertificateException {
        CertificateFactory cf = CertificateFactory.getInstance("X509");
        PublicKey publicKey = kp.getPublic();
        PrivateKey privateKey = kp.getPrivate();
        byte[] version = Encoder.encodeConstructed((byte) 0,
Encoder.encodePositiveBigInteger(new BigInteger("2"))); // version 1
        byte[] serialNo = Encoder.encodePositiveBigInteger(new BigInteger(1,
Util.computeKCV(publicKey.getEncoded())));

        // Use the SHA512 OID and algorithm.
        byte[] signatureOid = new byte[] {
            (byte) 0x2A, (byte) 0x86, (byte) 0x48, (byte) 0x86, (byte) 0xF7, (byte)
0x0D, (byte) 0x01, (byte) 0x01, (byte) 0x0D };
        String sigAlgoName = "SHA512WithRSA";

        byte[] signatureId = Encoder.encodeSequence(
            Encoder.encodeOid(signatureOid),
            Encoder.encodeNull());

        byte[] issuer = Encoder.encodeSequence(
            encodeName(COUNTRY_NAME_OID, "<Country>"),
            encodeName(STATE_OR_PROVINCE_NAME_OID, "<State>"),
            encodeName(LOCALITY_NAME_OID, "<City>"),
            encodeName(ORGANIZATION_NAME_OID,
"<Organization>"),
            encodeName(ORGANIZATION_UNIT_OID, "<Unit>"),
            encodeName(COMMON_NAME_OID, "<CN>")
        );
    }

```

```

    Calendar c = Calendar.getInstance();
    c.add(Calendar.DAY_OF_YEAR, -1);
    Date notBefore = c.getTime();
    c.add(Calendar.YEAR, 1);
    Date notAfter = c.getTime();
    byte[] validity = Encoder.encodeSequence(
        Encoder.encodeUTCTime(notBefore),
        Encoder.encodeUTCTime(notAfter)
    );

    byte[] key = publicKey.getEncoded();

    byte[] certificate = Encoder.encodeSequence(
        version,
        serialNo,
        signatureId,
        issuer,
        validity,
        issuer,
        key);

    Signature sig;
    byte[] signature = null;
    try {
        sig = Signature.getInstance(sigAlgoName, "Cavium");
        sig.initSign(privateKey);
        sig.update(certificate);
        signature = Encoder.encodeBitstring(sig.sign());

    } catch (Exception e) {
        System.err.println(e.getMessage());
        return null;
    }

    byte [] x509 = Encoder.encodeSequence(
        certificate,
        signatureId,
        signature
    );

    return cf.generateCertificate(new ByteArrayInputStream(x509));
}

//
// Simple OID encoder.
// Encode a value with OID in ASN.1 format

```

```
//
private static byte[] encodeName(byte[] name0id, String value) {
    byte[] name = null;
    name = Encoder.encodeSet(
        Encoder.encodeSequence(
            Encoder.encode0id(name0id),
            Encoder.encodePrintableString(value)
        )
    );
    return name;
}

//
// List all the keys in the keystore.
//
private static void listKeys(String keystoreFile, String password) throws Exception
{
    KeyStore keyStore = KeyStore.getInstance("CloudHSM");

    try {
        FileInputStream instream = new FileInputStream(keystoreFile);
        keyStore.load(instream, password.toCharArray());
    } catch (FileNotFoundException ex) {
        System.err.println("Keystore not found, loading an empty store");
        keyStore.load(null, null);
    }

    for(Enumeration<String> entry = keyStore.aliases(); entry.hasMoreElements();) {
        System.out.println(entry.nextElement());
    }
}
}
```

Use a Ferramenta de Geração e Edição de Manifestos da Microsoft (Mage.exe) com AWS CloudHSM para assinar arquivos

Note

AWS CloudHSM suporta somente a ferramenta Mage de 64 bits incluída no SDK do Windows para .NET Framework 4.8.1 e versões posteriores.

Os tópicos a seguir fornecem uma visão geral de como usar [o Mage.exe](#) com AWS CloudHSM.

Tópicos

- [Etapa 1: configurar os pré-requisitos](#)
- [Etapa 2: criar um certificado de assinatura](#)
- [Etapa 3: assinar um arquivo](#)

Etapa 1: configurar os pré-requisitos

Para usar o Microsoft Mage.exe com AWS CloudHSM, você precisa do seguinte:

- Uma EC2 instância da Amazon executando um sistema operacional Windows
- Uma autoridade de certificação (CA), auto-mantida ou estabelecida por um provedor terceirizado
- Um AWS CloudHSM cluster ativo na mesma nuvem privada virtual (VPC) da sua EC2 instância, com pelo menos um HSM
- Um usuário criptográfico (UC) para possuir e gerenciar chaves no cluster AWS CloudHSM
- Um arquivo não assinado ou executável
- O Kit de desenvolvimento de software (SDK) do Microsoft Windows

Para configurar os pré-requisitos para uso com o Mage.exe AWS CloudHSM

1. Execute uma EC2 instância do Windows e um AWS CloudHSM cluster seguindo as instruções na seção [Introdução](#) deste guia.
2. Se você quiser hospedar sua própria CA do Windows Server, conclua as etapas 1 e 2 em [Configurando o Windows Server como uma autoridade de certificação com AWS CloudHSM](#). Caso contrário, use a sua CA terceirizada confiável publicamente.

3. Baixe e instale o SDK do Microsoft Windows para .NET Framework 4.8.1 ou posterior em sua EC2 instância do Windows:
 - [SDK 10 do Microsoft Windows](#)

O executável `mage.exe` faz parte das ferramentas do SDK do Windows. O local de instalação padrão é:

```
C:\Program Files (x86)\Windows Kits\<SDK version>\bin\<version number>\x64\Mage.exe
```

Depois de concluir essas etapas, você pode usar o SDK do Microsoft Windows, seu AWS CloudHSM cluster e sua CA para [criar um certificado de assinatura](#).

Etapa 2: criar um certificado de assinatura

Agora que você instalou o SDK do Windows na sua EC2 instância, pode usá-lo para gerar uma solicitação de assinatura de certificado (CSR). A CSR é um certificado não assinado que você envia para a CA para assinatura. Neste exemplo, usamos o executável `certreq` incluído com o SDK do Windows para gerar a CSR.

Para gerar uma CSR usando o executável `certreq`

1. Conecte-se à sua EC2 instância do Windows. Para obter mais informações, consulte [Connect to Your Instance](#) no Guia EC2 do usuário da Amazon.
2. Crie um arquivo chamado `request.inf` com o conteúdo a seguir. Substitua as informações de `Subject` pelos detalhes da sua organização:

```
[Version]
Signature= $Windows NT$
[NewRequest]
Subject = "C=<Country>,CN=<www.website.com>,O=<Organization>,OU=<Organizational-Unit>,L=<City>,S=<State>"
RequestType=PKCS10
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "CloudHSM Key Storage Provider"
KeyUsage = "CERT_DIGITAL_SIGNATURE_KEY_USAGE"
MachineKeySet = True
```

```
Exportable = False
```

Para obter uma explicação de cada parâmetro, consulte a [Documentação da Microsoft](#).

3. Execute `certreq.exe` para gerar a CSR:

```
certreq.exe -new request.inf request.csr
```

Esse comando gera um novo par de chaves em seu AWS CloudHSM cluster e usa a chave privada para criar a CSR.

4. Envie a CSR à sua CA. Se estiver usando uma CA do Windows Server, siga estas etapas:

a. Abra a ferramenta de CA:

```
certsrv.msc
```

- b. Na nova janela, clique com o botão direito do mouse no nome do servidor da CA. Escolha All Tasks (Todas as tarefas) e depois escolha Submit new request (Enviar nova solicitação).
- c. Navegue até o local de `request.csr` e escolha Abrir.
- d. Navegue até a pasta Solicitações pendentes expandindo o menu CA do servidor. Clique com o botão direito do mouse na solicitação que você acabou de criar e, em Todas as tarefas, escolha Problema.
- e. Acesse a pasta Certificados emitidos.
- f. Escolha Open (Abrir) para visualizar o certificado e depois escolha a guia Details (Detalhes).
- g. Escolha Copy to File (Copiar no arquivo) para iniciar o Certificate Export Wizard (Assistente de exportação da CA). Salve o arquivo X.509 codificado por DER em um local seguro como `signedCertificate.cer`.
- h. Saia da ferramenta de CA e execute o comando a seguir para mover o arquivo do certificado para o armazenamento de certificados pessoais no Windows.

```
certreq.exe -accept signedCertificate.cer
```

Agora, você pode usar o certificado importado para [assinar um arquivo](#).

Etapa 3: assinar um arquivo

Agora que tem a Mage.exe e seu certificado importado, você pode assinar um arquivo. Você precisa saber o hash SHA-1 do certificado ou a impressão digital. A impressão digital garante que a Mage.exe use apenas certificados verificados por AWS CloudHSM. Neste exemplo, usamos PowerShell para obter o hash do certificado.

Para obter uma impressão digital do certificado e usá-lo para assinar um arquivo

1. Acesse o diretório que contém mage.exe. O local padrão é:

```
C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A\bin\NETFX 4.8.1 Tools\x64
```

2. Para criar um arquivo de aplicação de amostra usando Mage.exe, execute o seguinte comando:

```
mage.exe -New Application -ToFile C:\Users\Administrator\Desktop\sample.application
```

3. Abra PowerShell como administrador e execute o seguinte comando:

```
Get-ChildItem -path cert:\LocalMachine\My
```

Copie os valores Thumbprint, Key Container e Provider do resultado.

```
===== Certificate 1 =====  
Serial Number: 7ad8df4f5bfaf5a3405d32aa4a01b85a  
Issuer: CN=EC2AMAZ-0BVI3HH-CA  
NotBefore: 2/4/2025 7:55 PM  
NotAfter: 2/4/2030 8:05 PM  
Subject: CN=EC2AMAZ-0BVI3HH-CA  
CA Version: V0.0  
Signature matches Public Key  
Root Certificate: Subject matches Issuer  
Cert Hash(sha1): 08f295934325ee2c37ea01ab605ceeb81610d3be thumbprint  
Key Container = EC2AMAZ-0BVI3HH-CA key container  
Provider = CloudHSM Key Storage Provider provider  
Signature test passed  
CertUtil: -store command completed successfully.
```

4. Assine seu arquivo executando o seguinte comando:

```
mage.exe -Sign -CertHash <thumbprint> -KeyContainer <keycontainer> -  
CryptoProvider <CloudHSM Key Storage Provider/Cavium Key Storage Provider> C:\Users  
\Administrator\Desktop\<sample.application>
```

Se o comando for bem-sucedido, PowerShell retornará uma mensagem de sucesso.

5. Para verificar a assinatura no arquivo, use o seguinte comando:

```
mage.exe -Verify -CryptoProvider <CloudHSM Key Storage Provider/Cavium Key Storage Provider> C:\Users\Administrator\Desktop\<sample.application>
```

Outras integrações de fornecedores terceirizados com AWS CloudHSM

Vários fornecedores terceirizados oferecem suporte AWS CloudHSM como uma raiz de confiança. Assim, você pode utilizar a solução de software da sua preferência para criar e armazenar chaves subjacentes no seu cluster do CloudHSM. Como resultado, sua carga de trabalho AWS pode contar com os benefícios de latência, disponibilidade, confiabilidade e elasticidade do CloudHSM. A lista a seguir inclui fornecedores terceiros que oferecem suporte ao CloudHSM.

Note

AWS não endossa nem garante nenhum fornecedor terceirizado.

- A [Hashicorp Vault](#) é uma ferramenta de gerenciamento de segredos criada para permitir a colaboração e a governança entre organizações. Ele apóia AWS Key Management Service e AWS CloudHSM como raízes de confiança para proteção adicional.
- O [Thycotic Secrets Server](#) ajuda os clientes a gerenciar credenciais confidenciais para contas com base em privilégios. Ele se apóia AWS CloudHSM como uma raiz de confiança.
- O [adaptador KMIP do P6R](#) permite que você utilize suas AWS CloudHSM instâncias por meio de uma interface KMIP padrão.
- PrimeKey O [EJBCA](#) é uma solução popular de código aberto para PKI. Ele permite que você crie e armazene pares de chaves com segurança com. AWS CloudHSM
- KeySafeA [Box](#) fornece gerenciamento de chaves de criptografia para conteúdo em nuvem para muitas organizações com requisitos rígidos de segurança, privacidade e conformidade regulatória. Os clientes podem ainda proteger KeySafe as chaves diretamente AWS Key Management Service ou indiretamente AWS CloudHSM por meio do AWS KMS Custom Key Store.

- O [Insyde Software](#) oferece suporte AWS CloudHSM como raiz de confiança para assinatura de firmware.
- O [F5 BIG-IP LTM](#) oferece suporte AWS CloudHSM como uma raiz de confiança.
- O [Cloudera Navigator Key HSM](#) permite usar seu cluster do CloudHSM para criar e armazenar chaves para o Cloudera Navigator Key Trustee Server.
- A [Venafi Trust Protection Platform](#) fornece gerenciamento abrangente de identidade de máquina para TLS, SSH e assinatura de código com geração e proteção de chaves do AWS CloudHSM.

Monitoramento AWS CloudHSM

Além dos recursos de registro incorporados ao SDK do cliente, você também pode usar o AWS CloudTrail Amazon CloudWatch Logs e o Amazon CloudWatch para monitorar AWS CloudHSM.

Logs do Client SDK

Use o registro do SDK do cliente para monitorar as informações de diagnóstico e solução de problemas dos aplicativos que você cria.

CloudTrail

Use CloudTrail para monitorar todas as chamadas de API em sua AWS conta, incluindo as chamadas que você faz para criar e excluir clusters, módulos de segurança de hardware (HSM) e tags de recursos.

CloudWatch Registros

Use CloudWatch os registros para monitorar os registros de suas instâncias do HSM, que incluem eventos para criar e excluir usuários do HSM, alterar senhas de usuários, criar e excluir chaves e muito mais.

CloudWatch

Use CloudWatch para monitorar a integridade do seu cluster em tempo real.

Tópicos

- [Trabalhando com registros AWS CloudHSM do SDK do cliente](#)
- [Trabalhando com AWS CloudTrail e AWS CloudHSM](#)
- [Trabalhando com Amazon CloudWatch Logs e AWS CloudHSM Audit Logs](#)
- [Obtendo CloudWatch métricas para AWS CloudHSM](#)

Trabalhando com registros AWS CloudHSM do SDK do cliente

Você pode recuperar registros gerados pelo SDK do cliente. AWS CloudHSM oferece uma implementação de registro com o Client SDK 3 e o Client SDK 5.

Tópicos

- [Registro em log do Client SDK 5](#)
- [Registro em log do Client SDK 3](#)

Registro em log do Client SDK 5

Os logs do SDK 5 do cliente contêm informações de cada componente em um arquivo com o nome do componente. É possível usar a ferramenta de configuração do Client SDK 5 para configurar o registro em log para cada componente.

Se você não especificar um local para o arquivo, o sistema gravará os registros no local padrão:

AWS CloudHSM CLI

- Linux

```
/opt/cloudhsm/run/cloudhsm-cli.log
```

Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-cli.log
```

PKCS #11 library

- Linux

```
/opt/cloudhsm/run/cloudhsm-pkcs11.log
```

Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-pkcs11.log
```

OpenSSL Dynamic Engine

- Linux

```
stderr
```

JCE provider

- Linux

```
/opt/cloudhsm/run/cloudhsm-jce.log
```

Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-jce.log
```

Key Storage Provider

- Windows

```
C:\Program Files\Amazon\CloudHSM\cloudhsm-ksp.log
```

OpenSSL Dynamic Engine Provider

- Linux

```
stderr
```

Para obter informações sobre como configurar o registro em log para o Client SDK 5, consulte a ferramenta de [Configuração do Client SDK 5](#)

Registro em log do Client SDK 3

Os registros do SDK 3 do cliente contêm informações detalhadas do daemon do AWS CloudHSM cliente. A localização dos registros depende do sistema operacional da instância EC2 cliente da Amazon em que você executa o daemon do cliente.

Amazon Linux

No Amazon Linux, os registros AWS CloudHSM do cliente são gravados no arquivo chamado `/opt/cloudhsm/run/cloudhsm_client.log`. É possível usar o `logrotate` ou uma ferramenta similar para alternar e gerenciar esses logs.

Amazon Linux 2

No Amazon Linux 2, os registros AWS CloudHSM do cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

CentOS 7

No CentOS 7, os registros do AWS CloudHSM cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

CentOS 8

No CentOS 8, os registros do AWS CloudHSM cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

RHEL 7

No Red Hat Enterprise Linux 7, os registros AWS CloudHSM do cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

RHEL 8

No Red Hat Enterprise Linux 8, os registros AWS CloudHSM do cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

Ubuntu 16.04

No Ubuntu 16.04, os registros AWS CloudHSM do cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

Ubuntu 18.04

No Ubuntu 18.04, os registros AWS CloudHSM do cliente são coletados e armazenados no diário. É possível usar `journalctl` para visualizar e gerenciar esses logs. Por exemplo, use o comando a seguir para visualizar os registros AWS CloudHSM do cliente.

```
journalctl -f -u cloudhsm-client
```

Windows

- Para clientes Windows 1.1.2+:

AWS CloudHSM os registros do cliente são gravados em um `cloudhsm-kps.log` arquivo na pasta de arquivos do AWS CloudHSM programa (C:\Program Files\Amazon\CloudHSM \). Cada nome de arquivo de log é sufixado com um carimbo de data/hora indicando quando o AWS CloudHSM cliente foi iniciado.

- Para o cliente Windows 1.1.1 e anterior:

Os logs do cliente não são gravados em um arquivo. Os registros são exibidos no prompt de comando ou na PowerShell janela em que você iniciou o AWS CloudHSM cliente.

Trabalhando com AWS CloudTrail e AWS CloudHSM

AWS CloudHSM é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço em AWS CloudHSM. CloudTrail captura todas as chamadas de API AWS CloudHSM como eventos. As chamadas capturadas incluem chamadas do AWS CloudHSM console e chamadas de código para as operações AWS CloudHSM da API. Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para. AWS CloudHSM Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita AWS

CloudHSM, 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](#). Para ver uma lista completa das operações da AWS CloudHSM API, consulte [Ações](#) na Referência AWS CloudHSM da API.

AWS CloudHSM informações em CloudTrail

CloudTrail é ativado em sua AWS conta quando você cria a conta. Quando a atividade ocorre em AWS CloudHSM, essa atividade é registrada em um CloudTrail evento junto com outros eventos AWS de 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 AWS CloudHSM, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela 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 para Criar uma Trilha](#)
- [CloudTrail Serviços e integrações compatíveis](#)
- [Configurando notificações do Amazon SNS para CloudTrail](#)
- [Recebendo arquivos de CloudTrail log de várias regiões](#) e [recebendo arquivos de CloudTrail log de várias contas](#)

CloudTrail registra todas as AWS CloudHSM operações, incluindo operações somente para leitura, como `DescribeClusters` e `ListTags`, e operações de gerenciamento, como `InitializeClusterCreatHsm`, e `DeleteBackup`

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](#).

Entendendo as entradas do arquivo de AWS CloudHSM log

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 AWS CloudHSM `CreateHsm` ação.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AR0AJZVM5NEGZSTCITAMM:ExampleSession",
    "arn": "arn:aws:sts::111122223333:assumed-role/AdminRole/ExampleSession",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIY22AX6VRYNBJSA",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2017-07-11T03:48:44Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AR0AJZVM5NEGZSTCITAMM",
        "arn": "arn:aws:iam::111122223333:role/AdminRole",
        "accountId": "111122223333",
        "userName": "AdminRole"
      }
    }
  },
  "eventTime": "2017-07-11T03:50:45Z",
```

```

    "eventSource": "cloudhsm.amazonaws.com",
    "eventName": "CreateHsm",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.179",
    "userAgent": "aws-internal/3",
    "requestParameters": {
        "availabilityZone": "us-west-2b",
        "clusterId": "cluster-fw7mh6mayb5"
    },
    "responseElements": {
        "hsm": {
            "eniId": "eni-65338b5a",
            "clusterId": "cluster-fw7mh6mayb5",
            "state": "CREATE_IN_PROGRESS",
            "eniIp": "10.0.2.7",
            "hsmId": "hsm-6lz2hfmzbx",
            "subnetId": "subnet-02c28c4b",
            "availabilityZone": "us-west-2b"
        }
    },
    "requestID": "1dae0370-65ec-11e7-a770-6578d63de907",
    "eventID": "b73a5617-8508-4c3d-900d-aa8ac9b31d08",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}

```

Trabalhando com Amazon CloudWatch Logs e AWS CloudHSM Audit Logs

Quando um HSM em sua conta recebe um comando das [ferramentas da linha de AWS CloudHSM comando](#) ou das [bibliotecas de software](#), ele registra a execução do comando no formato de registro de auditoria. Os logs de auditoria do HSM incluem todos os [comandos de gerenciamento](#) iniciados pelo cliente, inclusive os que criam e excluem o HSM, fazem login e logout do HSM e gerenciam usuários e chaves. Esses logs fornecem um registro confiável de ações que foram alterados no estado do HSM.

AWS CloudHSM coleta seus registros de auditoria do HSM e os envia para a [Amazon CloudWatch Logs](#) em seu nome. Você pode usar os recursos do CloudWatch Logs para gerenciar seus registros de AWS CloudHSM auditoria, incluindo pesquisar e filtrar os logs e exportar dados de log para o

Amazon S3. Você pode trabalhar com seus registros de auditoria do HSM no [CloudWatch console da Amazon](#) ou usar os comandos CloudWatch Logs no [AWS CLI](#) e [CloudWatch Logs SDKs](#).

Tópicos

- [Como funciona o registro em log de auditoria do HSM](#)
- [Visualização AWS CloudHSM de registros de auditoria em CloudWatch Registros](#)
- [Interpretação de registros de AWS CloudHSM auditoria](#)
- [AWS CloudHSM referência do registro de auditoria](#)

Como funciona o registro em log de auditoria do HSM

O registro de auditoria é ativado automaticamente em todos os AWS CloudHSM clusters. Ele não pode ser desativado ou desativado, e nenhuma configuração pode AWS CloudHSM impedir a exportação dos registros para o CloudWatch Logs. Cada evento de log tem um time stamp e um número sequencial que indicam a ordem dos eventos e ajudam a detectar qualquer violação de logs.

Cada instância do HSM gera seu próprio log. É provável que os registros de auditoria de vários HSMs, mesmo aqueles no mesmo cluster, sejam diferentes. Por exemplo, apenas o primeiro HSM em cada cluster registra a inicialização do HSM. Os eventos de inicialização não aparecem nos registros HSMs que são clonados dos backups. Da mesma forma, quando você cria uma chave, o HSM que gera a chave registra um evento de geração de chave. Os outros HSMs no cluster registram um evento quando recebem a chave por meio de sincronização.

AWS CloudHSM coleta os registros e os publica em CloudWatch Logs da sua conta. Para se comunicar com o serviço de CloudWatch registros em seu nome, AWS CloudHSM use uma função [vinculada ao serviço](#). A política do IAM associada à função permite AWS CloudHSM realizar somente as tarefas necessárias para enviar os registros de auditoria para a CloudWatch Logs.

Important

Se tiver criado um cluster antes de 20 de janeiro de 2018 e ainda não tiver criado uma função vinculada ao serviço, você deverá criar uma manualmente. Isso é necessário CloudWatch para receber registros de auditoria do seu AWS CloudHSM cluster. Para obter mais informações sobre a criação da função vinculada ao serviço, consulte [Noções básicas das funções vinculadas ao serviço](#), bem como [Criar uma função vinculada ao serviço](#) no Guia do usuário do IAM.

Visualização AWS CloudHSM de registros de auditoria em CloudWatch Registros

O Amazon CloudWatch Logs organiza os registros de auditoria em grupos de registros e, dentro de um grupo de registros, em fluxos de registros. Cada entrada de registro é um evento. AWS CloudHSM cria um grupo de registros para cada cluster e um fluxo de registros para cada HSM no cluster. Você não precisa criar nenhum componente do CloudWatch Logs nem alterar nenhuma configuração.

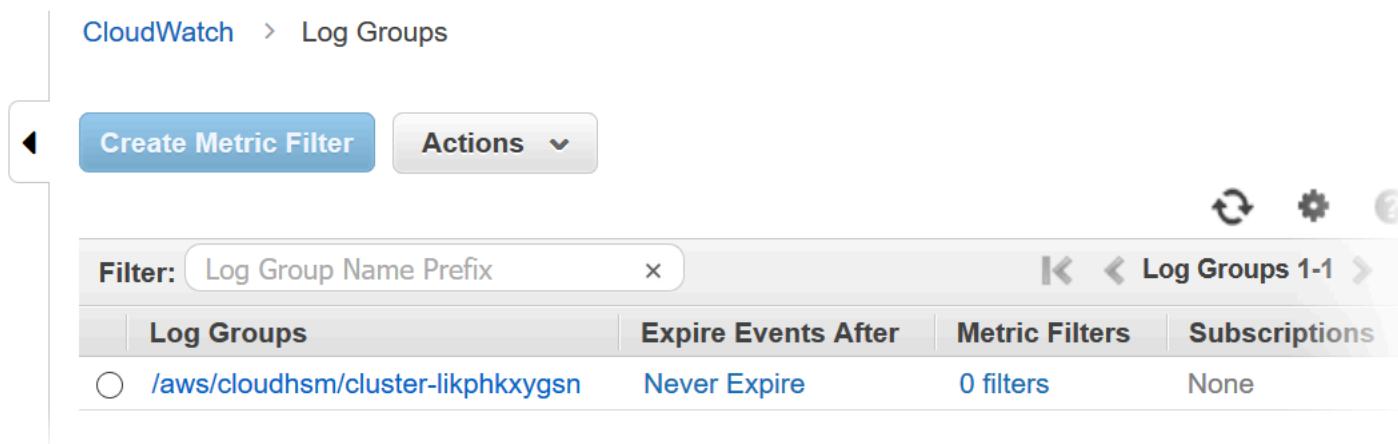
- O nome do grupo de logs é `/aws/cloudhsm/<cluster ID>`; por exemplo, `/aws/cloudhsm/cluster-likphkxygsn`. Ao usar o nome do grupo de registros em um PowerShell comando AWS CLI or, certifique-se de colocá-lo entre aspas duplas.
- O nome do fluxo de logs é o ID do HSM; por exemplo, `hsm-nwbbiqb4jk`.

Em geral, há um fluxo de logs para cada HSM. No entanto, qualquer ação que altera o ID do HSM, como quando um HSM falha e é substituída, cria um novo fluxo de logs.

Para obter mais informações sobre CloudWatch os conceitos de registros, consulte [Conceitos](#) no Guia do usuário do Amazon CloudWatch Logs.

[Você pode visualizar os registros de auditoria de um HSM na página CloudWatch Logs Console de gerenciamento da AWS, nos comandos CloudWatch Logs no AWS CLI, nos PowerShellcmdlets CloudWatch Logs ou nos Logs. CloudWatch SDKs](#) Para obter instruções, consulte [Exibir dados de registro](#) no Guia do usuário do Amazon CloudWatch Logs.

Por exemplo, a imagem a seguir mostra o grupo de logs do cluster do `cluster-likphkxygsn` no Console de gerenciamento da AWS.



Ao escolher o nome do grupo de logs do cluster, você pode visualizar o fluxo de log de cada um HSMs no cluster. A imagem a seguir mostra os fluxos de log do HSMs no `cluster-likphkxygsn` cluster.

CloudWatch > Log Groups > Streams for /aws/cloudhsm/cluster-likphkxygsn

Search Log Group Create Log Stream Delete Log Stream

Filter: Log Stream Name Prefix x

Log Streams 1-2

☐	Log Streams	Last Event Time
☐	hsm-aht4p3sgs3c	2017-12-28 06:12 UTC-8
☐	hsm-xkvjp4wk5o3	2017-12-28 06:12 UTC-8

Ao escolher um nome do fluxo de logs do HSM, você pode visualizar os eventos no log de auditoria. Por exemplo, este evento, que tem um número de sequência de 0x0 e um Opcode de CN_INIT_TOKEN, geralmente é o primeiro evento do primeiro HSM em cada cluster. Ele registra a inicialização do HSM no cluster.

Filter events

Time (UTC +00:00)	Message
2017-12-19	Time: 12/19/17 21:01:16.962174, usecs:1513717276962174 Sequence No : 0x0 Reboot counter : 0xe8 Command Type(hex) : CN_MGMT_CMD (0x0) Opcode : CN_INIT_TOKEN (0x1) Session Handle : 0x1004001 Response : 0:HSM Return: SUCCESS Log type : MINIMAL_LOG_ENTRY (0)

Você pode usar todos os vários recursos do CloudWatch Logs para gerenciar seus registros de auditoria. Por exemplo, você pode usar o recurso Filtrar eventos para encontrar um texto em especial em um evento, como o CN_CREATE_USER Opcode.

Para encontrar todos os eventos que não incluam o texto especificado, adicione um sinal de subtração (-) antes do texto. Por exemplo, para encontrar eventos que não incluam CN_CREATE_USER, digite -CN_CREATE_USER.

CN_CREATE_USER	
Time (UTC +00:00)	Message
2017-12-20	
No older events	
▼ 00:04:53	Time: 12/20/17 00:04:53.635826, u
Time: 12/20/17 00:04:53.635826, usecs:1513728293635826 Sequence No : 0x13a Reboot counter : 0xe8 Command Type(hex) : CN_MGMT_CMD (0x0) Opcode : CN_CREATE_USER (0x3) Session Handle : 0x1014006 Response : 0:HSM Return: SUCCESS Log type : MGMT_USER_DETAILS_LOG (2) User Name : testuser User Type : CN_CRYPT_USER (1)	

Interpretação de registros de AWS CloudHSM auditoria

Os eventos nos logs de auditoria do HSM têm campos padrão. Alguns tipos de eventos têm campos adicionais que capturam informações úteis sobre o evento. Por exemplo, eventos de login de usuário e gerenciamento de usuários incluem o nome de usuário e o tipo do usuário. Os comandos de gerenciamento de chaves incluem o identificador de chaves.

Vários dos campos fornecem informações especialmente importantes. O Opcode identifica o comando de gerenciamento que está sendo registrado. O Sequence No identifica um evento no fluxo de logs e indica a ordem em que ele foi registrado.

Por exemplo, o evento de exemplo a seguir é o segundo evento (Sequence No: 0x1) no fluxo de logs para um HSM. Ele mostra o HSM gerando uma chave de criptografia da senha, que faz parte da rotina de inicialização.

```
Time: 12/19/17 21:01:17.140812, usecs:1513717277140812
Sequence No : 0x1
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_GEN_PSWD_ENC_KEY (0x1d)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

Os campos a seguir são comuns a todos os AWS CloudHSM eventos no registro de auditoria.

Hora

A hora em que o evento ocorreu no fuso horário UTC. A hora é exibida como uma hora legível e hora Unix em microssegundos.

Contador de reinicializações

Um contador ordinal persistente de 32 bits incrementado quando o hardware do HSM é reinicializado.

Todos os eventos em um fluxo de logs têm o mesmo valor do contador de reinicializações. No entanto, o contador de reinicializações pode não ser exclusivo de um fluxo de logs, porque ele pode ser diferente em diferentes instâncias do HSM no mesmo cluster.

Número da sequência

Um contador ordinal de 64 bits incrementado para cada evento de log. O primeiro evento em cada fluxo de logs tem um número de sequência de 0x0. Não deve haver espaço nos valores de Sequence No. O número de sequência só é exclusivo em um fluxo de logs.

Tipo de comando

Um valor hexadecimal que representa a categoria do comando. Comandos nos fluxos de logs do AWS CloudHSM têm um tipo de comando de CN_MGMT_CMD (0x0) ou CN_CERT_AUTH_CMD (0x9).

Opcode

Identifica o comando de gerenciamento executado. Para obter uma lista de Opcode valores nos registros AWS CloudHSM de auditoria, consulte [AWS CloudHSM referência do registro de auditoria](#).

Identificador da sessão

Identifica a sessão em que o comando foi executado e o evento foi registrado.

Resposta

Registra a resposta ao comando de gerenciamento. Você pode pesquisar os valores SUCCESS e ERROR no campo Response.

Tipo de log

Indica o tipo de AWS CloudHSM registro do registro que gravou o comando.

- MINIMAL_LOG_ENTRY (0)
- MGMT_KEY_DETAILS_LOG (1)
- MGMT_USER_DETAILS_LOG (2)
- GENERIC_LOG

Exemplos de eventos de log de auditoria

Os eventos em um fluxo de logs registram o histórico do HSM da criação à exclusão. Você pode usar o registro para revisar o ciclo de vida do seu HSMs e obter informações sobre sua operação. Quando você interpretar os eventos, observe o Opcode, que indica o comando de gerenciamento ou ação, e o Sequence No, que indica a ordem dos eventos.

Tópicos

- [Exemplo: inicializar o primeiro HSM em um cluster](#)
- [Eventos de login e logout](#)
- [Exemplo: criar e excluir usuários](#)
- [Exemplo: criar e excluir um par de chaves](#)
- [Exemplo: gerar e sincronizar uma chave](#)
- [Exemplo: exportar uma chave](#)
- [Exemplo: importar uma chave](#)
- [Exemplo: compartilhar e descompartilhar uma chave](#)

Exemplo: inicializar o primeiro HSM em um cluster

O fluxo de registros de auditoria do primeiro HSM em cada cluster difere significativamente dos fluxos de registros de outros HSMs no cluster. O log de auditoria do primeiro HSM em cada cluster registra a criação e a inicialização. Os registros adicionais HSMs no cluster, que são gerados a partir de backups, começam com um evento de login.

Important

As seguintes entradas de inicialização não aparecerão nos CloudWatch registros dos clusters inicializados antes do lançamento do recurso de registro de auditoria do CloudHSM (30 de agosto de 2018). Para obter mais informações, consulte [Histórico do documento](#).

Os eventos de exemplo a seguir aparecem no fluxo de logs do primeiro HSM em um cluster. O primeiro evento no log, aquele com Sequence No 0x0, representa o comando para inicializar o HSM (CN_INIT_TOKEN). A resposta indica que o comando foi bem-sucedido (Response : 0: HSM Return: SUCCESS).

```
Time: 12/19/17 21:01:16.962174, usecs:1513717276962174
Sequence No : 0x0
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INIT_TOKEN (0x1)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

O segundo evento neste fluxo de logs de exemplo (Sequence No 0x1) registra o comando para criar a chave de criptografia de senha que o HSM usa (CN_GEN_PSWD_ENC_KEY).

Esta é uma sequência de inicialização típica para o primeiro HSM em cada cluster. Como os subseqüentes HSMs no mesmo cluster estão clones do primeiro, eles usam a mesma chave de criptografia de senha.

```
Time: 12/19/17 21:01:17.140812, usecs:1513717277140812
Sequence No : 0x1
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_GEN_PSWD_ENC_KEY (0x1d)
```

```
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

O terceiro evento neste fluxo de logs de exemplo (Sequence No 0x2) é a criação do [usuário do dispositivo \(AU\)](#), que é o serviço AWS CloudHSM . Os eventos que envolvem os usuários do HSM incluem campos extras para o nome do usuário e o tipo de usuário.

```
Time: 12/19/17 21:01:17.174902, usecs:1513717277174902
Sequence No : 0x2
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_CREATE_APPLIANCE_USER (0xfc)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : app_user
User Type : CN_APPLIANCE_USER (5)
```

O quarto evento neste fluxo de logs de exemplo (Sequence No 0x3) registra o evento CN_INIT_DONE, que conclui a inicialização do HSM.

```
Time: 12/19/17 21:01:17.298914, usecs:1513717277298914
Sequence No : 0x3
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INIT_DONE (0x95)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

Você pode seguir os demais eventos na sequência de inicialização. Esses eventos podem incluir vários eventos de login e de logout e a geração da chave de criptografia de chaves (KEK- key encryption key). O evento a seguir registra o comando que altera a senha do [responsável pela pré-criptografia \(PRECO\)](#). Esse comando ativa o cluster.

```
Time: 12/13/17 23:04:33.846554, usecs:1513206273846554
Sequence No: 0x1d
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_CHANGE_PSWD (0x9)
```

```
Session Handle: 0x2010003
Response: 0:HSM Return: SUCCESS
Log type: MGMT_USER_DETAILS_LOG (2)
User Name: admin
User Type: CN_CRYPT0_PRE_OFFICER (6)
```

Eventos de login e logout

Ao interpretar o log de auditoria, observe eventos que registram usuários que fazem login e logout do HSM. Esses eventos ajudam você a determinar qual usuário é responsável por comandos de gerenciamento que aparecem na sequência entre os comandos de login e logout.

Por exemplo, essa entrada de log registra um login por um responsável pela criptografia chamado admin. O número de sequência, 0x0, indica que esse é o primeiro evento neste fluxo de logs.

Quando um usuário faz login em um HSM, o outro HSMs no cluster também registra um evento de login para o usuário. Você pode encontrar os eventos de login correspondentes nos fluxos de log de outros HSMs no cluster logo após o evento de login inicial.

```
Time: 01/16/18 01:48:49.824999, usecs:1516067329824999
Sequence No : 0x0
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7014006
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : admin
User Type : CN_CRYPT0_OFFICER (2)
```

O evento de exemplo a seguir registra o logout do responsável pela criptografia admin. O número de sequência, 0x2, indica que esse é o terceiro evento no fluxo de logs.

Se o usuário conectado fechar a sessão sem fazer logout, o fluxo de logs incluirá um CN_APP_FINALIZE, ou evento de fechamento da sessão (CN_SESSION_CLOSE), em vez de um evento CN_LOGOUT. Diferentemente do evento de login, esse evento de logout normalmente só será registrado pelo HSM que executar o comando.

```
Time: 01/16/18 01:49:55.993404, usecs:1516067395993404
Sequence No : 0x2
Reboot counter : 0x107
```

```
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGOUT (0xe)
Session Handle : 0x7014000
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : admin
User Type : CN_CRYPT0_OFFICER (2)
```

Se uma tentativa de login falhar porque o nome do usuário é inválido, o HSM registrará um evento CN_LOGIN com o nome de usuário e o tipo fornecidos no comando de login. A resposta exibe a mensagem de erro 157, que explica que o nome do usuário não existe.

```
Time: 01/24/18 17:41:39.037255, usecs:1516815699037255
Sequence No : 0x4
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0xc008002
Response : 157:HSM Error: user isn't initialized or user with this name doesn't exist
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : ExampleUser
User Type : CN_CRYPT0_USER (1)
```

Se uma tentativa de login falhar porque a senha é inválida, o HSM registrará um evento CN_LOGIN com o nome de usuário e o tipo fornecidos no comando de login. A resposta exibe a mensagem de erro com o código de erro RET_USER_LOGIN_FAILURE.

```
Time: 01/24/18 17:44:25.013218, usecs:1516815865013218
Sequence No : 0x5
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0xc008002
Response : 163:HSM Error: RET_USER_LOGIN_FAILURE
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPT0_USER (1)
```

Exemplo: criar e excluir usuários

Este exemplo mostra os eventos de log registrados quando um responsável pela criptografia (CO) cria e exclui os usuários.

O primeiro evento registra um CO, admin, fazendo login no HSM. O número de sequência de 0x0 indica que esse é o primeiro evento no fluxo de logs. O nome e o tipo do usuário que se conectou são incluídos no evento.

```
Time: 01/16/18 01:48:49.824999, usecs:1516067329824999
Sequence No : 0x0
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7014006
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : admin
User Type : CN_CRYPT0_OFFICER (2)
```

O próximo evento na sequência de logs (sequência 0x1) registra o CO criando um novo usuário de criptografia (CU). O nome e o tipo do novo usuário são incluídos no evento.

```
Time: 01/16/18 01:49:39.437708, usecs:1516067379437708
Sequence No : 0x1
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_CREATE_USER (0x3)
Session Handle : 0x7014006
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : bob
User Type : CN_CRYPT0_USER (1)
```

Em seguida, o CO cria outro responsável pela criptografia, alice. O número de sequência indica que essa ação seguiu a anterior, sem ações de intervenção.

```
Time: 01/16/18 01:49:55.993404, usecs:1516067395993404
Sequence No : 0x2
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_CREATE_CO (0x4)
Session Handle : 0x7014007
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : alice
```

```
User Type : CN_CRYPT0_OFFICER (2)
```

Posteriormente, o CO chamado admin faz login e exclui o responsável pela criptografia chamado alice. O HSM registra um evento CN_DELETE_USER. O nome e o tipo do usuário excluído são incluídos no evento.

```
Time: 01/23/18 19:58:23.451420, usecs:1516737503451420
Sequence No : 0xb
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_DELETE_USER (0xa1)
Session Handle : 0x7014007
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : alice
User Type : CN_CRYPT0_OFFICER (2)
```

Exemplo: criar e excluir um par de chaves

Este exemplo mostra os eventos que são registrados em um log de auditoria do HSM ao criar e excluir um par de chaves.

O evento a seguir registra o usuário de criptografia (CU) chamado crypto_user fazendo login no HSM.

```
Time: 12/13/17 23:09:04.648952, usecs:1513206544648952
Sequence No: 0x28
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_LOGIN (0xd)
Session Handle: 0x2014005
Response: 0:HSM Return: SUCCESS
Log type: MGMT_USER_DETAILS_LOG (2)
User Name: crypto_user
User Type: CN_CRYPT0_USER (1)
```

Em seguida, o usuário gera um par de chaves (CN_GENERATE_KEY_PAIR). A chave privada possui o identificador de chave 131079. A chave pública possui o identificador de chave 131078.

```
Time: 12/13/17 23:09:04.761594, usecs:1513206544761594
Sequence No: 0x29
```

```
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_GENERATE_KEY_PAIR (0x19)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
Log type: MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle: 131079
Public Key Handle: 131078
```

O usuário exclui imediatamente o par de chaves. Um evento CN_DESTROY_OBJECT registra a exclusão da chave pública (131078).

```
Time: 12/13/17 23:09:04.813977, usecs:1513206544813977
Sequence No: 0x2a
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_DESTROY_OBJECT (0x11)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
Log type: MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle: 131078
Public Key Handle: 0
```

Em seguida, um segundo evento CN_DESTROY_OBJECT registra a exclusão da chave privada (131079).

```
Time: 12/13/17 23:09:04.815530, usecs:1513206544815530
Sequence No: 0x2b
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_DESTROY_OBJECT (0x11)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
Log type: MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle: 131079
Public Key Handle: 0
```

Por fim, o usuário faz logout.

```
Time: 12/13/17 23:09:04.817222, usecs:1513206544817222
Sequence No: 0x2c
```



```
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_LOGOUT (0xe)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
Log type: MGMT_USER_DETAILS_LOG (2)
User Name: crypto_user
User Type: CN_CRYPT0_USER (1)
```

Exemplo: gerar e sincronizar uma chave

Este exemplo mostra o efeito da criação de uma chave em um cluster com várias HSMs. A chave é gerada em um HSM, extraída do HSM como um objeto mascarado e inserida no outro HSMs como um objeto mascarado.

Note

As ferramentas de cliente podem falhar ao sincronizar a chave. Ou o comando pode incluir o `min_srv` parâmetro, que sincroniza a chave somente com o número especificado de HSMs. Em ambos os casos, o AWS CloudHSM serviço sincroniza a chave com a outra HSMs no cluster. Como eles HSMs registram somente comandos de gerenciamento do lado do cliente em seus registros, a sincronização do lado do servidor não é registrada no registro do HSM.

Primeiro considere o fluxo de logs do HSM que recebe e executa os comandos. O fluxo de logs é nomeado de acordo com o ID do HSM, `hsm-abcde123456`, mas o ID do HSM não aparece nos eventos de log.

Primeiro, o usuário de criptografia (CU) `testuser` faz login no HSM do `hsm-abcde123456`.

```
Time: 01/24/18 00:39:23.172777, usecs:1516754363172777
Sequence No : 0x0
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0xc008002
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPT0_USER (1)
```

A UC executa um [exSymKey](#) comando para gerar uma chave simétrica. O HSM hsm-abcde123456 gera uma chave simétrica com um identificador de chaves de 262152. O HSM registra um evento CN_GENERATE_KEY em seu log.

```
Time: 01/24/18 00:39:30.328334, usecs:1516754370328334
Sequence No : 0x1
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_GENERATE_KEY (0x17)
Session Handle : 0xc008004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 262152
Public Key Handle : 0
```

O próximo evento no fluxo de logs de hsm-abcde123456 registra a primeira etapa no processo de sincronização de chaves. A nova chave (identificador de chaves 262152) é extraída do HSM como um objeto mascarado.

```
Time: 01/24/18 00:39:30.330956, usecs:1516754370330956
Sequence No : 0x2
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_EXTRACT_MASKED_OBJECT_USER (0xf0)
Session Handle : 0xc008004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 262152
Public Key Handle : 0
```

Agora, considere o fluxo de logs do HSM hsm-zyxwv987654, outro HSM no mesmo cluster. Esse fluxo de logs também inclui um evento de login do usuário testuser. O valor de tempo mostra que ocorre logo depois que o usuário faz login no HSM hsm-abcde123456.

```
Time: 01/24/18 00:39:23.199740, usecs:1516754363199740
Sequence No : 0xd
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7004004
Response : 0:HSM Return: SUCCESS
```

```
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPT0_USER (1)
```

Esse fluxo de logs desse HSM não tem um evento CN_GENERATE_KEY. No entanto, ele não possui um evento que registra a sincronização da chave desse HSM. O evento CN_INSERT_MASKED_OBJECT_USER registra o recebimento da chave 262152 como um objeto mascarado. Agora, a chave 262152 existe HSMs em ambos no cluster.

```
Time: 01/24/18 00:39:30.408950, usecs:1516754370408950
Sequence No : 0xe
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INSERT_MASKED_OBJECT_USER (0xf1)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 262152
Public Key Handle : 0
```

Quando o usuário faz login, esse evento CN_LOGOUT é exibido somente no fluxo de logs do HSM que recebeu os comandos.

Exemplo: exportar uma chave

Este exemplo mostra os eventos do log de auditoria que são registrados quando um usuário criptográfico (UC) exporta chaves de um cluster com várias HSMs.

O evento a seguir registra o CU (testuser) fazendo login no [key_mgmt_util](#).

```
Time: 01/24/18 19:42:22.695884, usecs:1516822942695884
Sequence No : 0x26
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7004004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPT0_USER (1)
```

A UC executa um [exSymKey](#) comando para exportar a chave7, uma chave AES de 256 bits. O comando usa key6, uma chave AES de 256 bits no HSMs, como chave de empacotamento.

O HSM que recebe o comando registra um evento CN_WRAP_KEY para a chave 7, a chave que está sendo exportada.

```
Time: 01/24/18 19:51:12.860123, usecs:1516823472860123
Sequence No : 0x27
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_WRAP_KEY (0x1a)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 7
Public Key Handle : 0
```

Em seguida, o HSM registra um evento CN_NIST_AES_WRAP para a chave de encapsulamento, a chave 6. A chave é encapsulada e, em seguida, desencapsulada imediatamente, mas o HSM registra apenas um evento.

```
Time: 01/24/18 19:51:12.905257, usecs:1516823472905257
Sequence No : 0x28
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_NIST_AES_WRAP (0x1e)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 6
Public Key Handle : 0
```

O comando exSymKey grava a chave exportada em um arquivo, mas não altera a chave no HSM. Consequentemente, não há eventos correspondentes nos registros de outros HSMs no cluster.

Exemplo: importar uma chave

Este exemplo mostra os eventos do log de auditoria que são registrados quando você importa chaves para o HSMs em um cluster. Neste exemplo, o usuário criptográfico (CU) usa o [imSymKey](#) comando para importar uma chave AES para o. HSMs O comando usa a chave 6 como a chave de encapsulamento.

O HSM que recebe os comandos primeiro registra um evento CN_NIST_AES_WRAP para a chave 6, a chave de encapsulamento.

```
Time: 01/24/18 19:58:23.170518, usecs:1516823903170518
Sequence No : 0x29
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_NIST_AES_WRAP (0x1e)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 6
Public Key Handle : 0
```

Em seguida, o HSM registra um evento CN_UNWRAP_KEY que representa a operação de importação. A chave importada é atribuída a um identificador de chaves de 11.

```
Time: 01/24/18 19:58:23.200711, usecs:1516823903200711
Sequence No : 0x2a
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_UNWRAP_KEY (0x1b)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 11
Public Key Handle : 0
```

Quando uma nova chave é gerada ou importada, as ferramentas do cliente tentam automaticamente sincronizar a nova chave com outras HSMs no cluster. Neste caso, o HSM registra um evento CN_EXTRACT_MASKED_OBJECT_USER quando a chave 11 é extraída do HSM como um objeto mascarado.

```
Time: 01/24/18 19:58:23.203350, usecs:1516823903203350
Sequence No : 0x2b
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_EXTRACT_MASKED_OBJECT_USER (0xf0)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
```

```
Priv/Secret Key Handle : 11
Public Key Handle : 0
```

Os fluxos de log de outros HSMs no cluster refletem a chegada da chave recém-importada.

Por exemplo, este evento foi registrado no fluxo de logs de um HSM diferente no mesmo cluster. Esse evento `CN_INSERT_MASKED_OBJECT_USER` registra a chegada de um objeto mascarado que representa a chave 11.

```
Time: 01/24/18 19:58:23.286793, usecs:1516823903286793
Sequence No : 0xb
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INSERT_MASKED_OBJECT_USER (0xf1)
Session Handle : 0xc008004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 11
Public Key Handle : 0
```

Exemplo: compartilhar e descompartilhar uma chave

Este exemplo mostra o evento de log de auditoria que é registrado quando um usuário de criptografia (CU) compartilha ou descompartilha a chave privada do ECC com outros usuários de criptografia. O CU usa o comando [shareKey](#) e fornece o identificador da chave, o ID de usuário e o valor 1 para compartilhar ou o valor 0 para descompartilhar a chave.

No exemplo a seguir, o HSM que recebe o comando, registra um evento `CM_SHARE_OBJECT` que representa a operação de compartilhamento.

```
Time: 02/08/19 19:35:39.480168, usecs:1549654539480168
Sequence No : 0x3f
Reboot counter : 0x38
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_SHARE_OBJECT (0x12)
Session Handle : 0x3014007
Response : 0:HSM Return: SUCCESS
Log type : UNKNOWN_LOG_TYPE (5)
```

AWS CloudHSM referência do registro de auditoria

AWS CloudHSM registra os comandos de gerenciamento do HSM em eventos de registro de auditoria. Cada evento tem um valor do código de operação (Opcode) que identifica a ação ocorrida e sua resposta. Você pode usar os valores Opcode para pesquisar, classificar e filtrar os logs.

A tabela a seguir define os Opcode valores em um registro AWS CloudHSM de auditoria.

Código de operação (Opcode)	Description
Login do usuário: esses eventos incluem o nome do usuário e o tipo de usuário	
CN_LOGIN (0xd)	Login do usuário
CN_LOGOUT (0xe)	Logout do usuário
CN_APP_FINALIZE	A conexão com o HSM foi encerrada. Todas as chaves de sessão ou tokens de quórum dessa conexão foram excluídos.
CN_CLOSE_SESSION	A sessão com o HSM foi encerrada. Todas as chaves de sessão ou tokens de quórum dessa sessão foram excluídos.
Gerenciamento de usuários: esses eventos incluem o nome do usuário e o tipo de usuário	
CN_CREATE_USER (0x3)	Criar um usuário de criptografia (CU)
CN_CREATE_CO	Criar um responsável pela criptografia (CO)
CN_DELETE_USER	Exclusão de um usuário
CN_CHANGE_PSWD	Alterar a senha de um usuário
CN_SET_M_VALUE	Definir autenticação de quorum (M de N) para uma ação do usuário
CN_APPROVE_TOKEN	Aprovar um token de autenticação de quórum para uma ação do usuário
CN_DELETE_TOKEN	Excluir um ou mais tokens de quórum

Código de operação (Opcode)	Description
CN_GET_TOKEN	Solicitar um token de assinatura para iniciar uma operação de quórum
Gerenciamento de chaves: esses eventos incluem o identificador de chaves	
CN_GENERATE_KEY	Gerar uma chave simétrica
CN_GENERATE_KEY_PAIR (0x19)	Gere um par de chaves assimétrico
CN_CREATE_OBJECT	Importar uma chave pública (sem encapsulamento)
CN_MODIFY_OBJECT	Definir um atributo-chave
CN_DESTROY_OBJECT (0x11)	Exclusão de uma chave de sessão
CN_TOMBSTONE_OBJECT	Exclusão de uma chave de token
CN_SHARE_OBJECT	Compartilhar ou descompartilhar uma chave
CN_WRAP_KEY	Exportar uma cópia criptografada de uma chave (wrapKey)
CN_UNWRAP_KEY	Importar uma cópia criptografada de uma chave (unwrapKey)
CN_DERIVE_KEY	Derivar uma chave simétrica de uma chave existente
CN_NIST_AES_WRAP	Criptografar ou descriptografar uma chave com uma chave AES
CN_INSERT_MASKED_OBJECT_USER	Insira uma chave criptografada com atributos de outro HSM no cluster.
CN_EXTRACT_MASKED_OBJECT_USER	Encapsula/criptografa uma chave com atributos do HSM para ser enviada para outro HSM no cluster.

Código de operação (Opcode)	Description
Session Management	
CN_ENCRYPT_SESSION_V2 (0x107)	Estabelece uma sessão end-to-end criptografada autenticada.
END_MARKER_OPCODE (0xffff)	Insere um marcador final no buffer de registros de auditoria indicando que não são permitidos mais comandos registráveis no HSM
Back up HSMs	
CN_BACKUP_BEGIN	Inicie o processo de backup
CN_BACKUP_END	Concluiu o processo de backup
CN_RESTORE_BEGIN	Comece a restaurar a partir de um backup
CN_RESTORE_END	Concluiu o processo de restauração a partir de um backup
Certificate-Based Authentication	
CN_CERT_AUTH_STORE_CERT	Armazena o certificado de cluster
HSM Instance Commands	
CN_INIT_TOKEN (0x1)	Inicie o processo de inicialização do HSM
CN_INIT_DONE	O processo de inicialização do HSM foi concluído
CN_GEN_KEY_ENC_KEY	Gerar uma Key Encryption Key (KEK - Chave de criptografia de chaves)
CN_GEN_PSWD_ENC_KEY (0x1d)	Gerar uma Password Encryption Key (PEK - Chave de criptografia de senhas)
HSM crypto commands	

Código de operação (Opcode)	Description
CN_FIPS_RAND	Gere um número aleatório compatível com FIPS ¹

[1] Só é registrado para clusters hsm1.medium.

Obtendo CloudWatch métricas para AWS CloudHSM

Use CloudWatch para monitorar seu AWS CloudHSM cluster em tempo real. As métricas podem ser agrupadas por região, pelo ID do cluster e pelo ID do HSM.

O namespace AWS/CloudHSM inclui as métricas a seguir:

Métrica	Description
HsmUnhealthy	A instância do HSM não está funcionando corretamente. AWS CloudHSM substitui automaticamente as instâncias não íntegras para você. Você pode optar por expandir proativamente o tamanho do cluster para reduzir o impacto no desempenho enquanto estamos substituindo o HSM.
HsmTemperature ¹	A temperatura de junção do processador de hardware. O sistema será desligado se a temperatura atingir 110 graus centígrados.
HsmKeysSessionOccupied	O número de chaves de sessão que estão sendo usadas pela instância do HSM.
HsmKeysTokenOccupied	O número de chaves de token que estão sendo usadas pela instância do HSM e pelo cluster.
HsmSslContextsOccupied ¹	O número de canais end-to-end criptografados atualmente estabelecidos para a instância do HSM. Até 2.048 canais são permitidos.
HsmSessionCount	O número de conexões abertas para a instância do HSM. Até 2.048 são permitidas. Por padrão, o daemon do cliente está configurado para abrir duas sessões com cada instância do HSM em um end-to-end canal criptografado.

Métrica	Description
	AWS CloudHSM também pode ter até 2 conexões abertas com o HSM para monitorar a integridade do HSMs.
HsmUsersAvailable	O número de usuários adicionais que podem ser criados. Isso é igual ao número máximo de usuários (listados em HsmUsersMax) menos os usuários criados até o momento.
HsmUsersMax ¹	HsmUsersMax: número máximo de usuários que podem ser criados na instância do HSM.
InterfaceEth2OctetsInput ¹	A soma cumulativa do tráfego de entrada no HSM até o momento.
InterfaceEth2OctetsOutput ¹	A soma cumulativa do tráfego de saída no HSM até o momento.

- [1] Essa métrica não está disponível para o tipo hsm2m.medium.

AWS CloudHSM informações de desempenho

Para AWS CloudHSM clusters de produção, você deve ter pelo menos duas instâncias do módulo de segurança de hardware (HSM) espalhadas por diferentes zonas de disponibilidade em uma região. Recomendamos testar a carga do seu cluster para determinar a carga máxima que você deve prever e, em seguida, adicionar mais um HSM a ele para garantir a alta disponibilidade. Para aplicativos que exigem durabilidade de chaves recém-geradas, recomendamos pelo menos três instâncias do HSM distribuídas em todas as zonas de disponibilidade de uma região.

Dados de desempenho

O desempenho dos AWS CloudHSM clusters varia com base na carga de trabalho específica. Para aumentar o desempenho, você pode adicionar outras instâncias de HSM aos seus clusters. O desempenho pode variar com base na configuração, no tamanho dos dados e na carga adicional do aplicativo em suas instâncias do EC2. Incentivamos o teste de carga de seu aplicativo para determinar as necessidades de escalabilidade.

A tabela a seguir mostra o desempenho aproximado de algoritmos criptográficos comuns executados em uma instância do EC2 com instâncias hsm1.medium.

Dados de desempenho para hsm1.medium

Operation	Cluster de dois HSM ¹	Cluster de três HSM ²	Cluster de seis HSM ³
Sinal RSA de 2048 bits	2.000 operações/seg	3.000 operações/seg	5.000 operações/seg
Sinal EC P256	500 operações/seg	750 operações/seg	1.500 operações/seg

A tabela abaixo mostra o desempenho aproximado dos algoritmos criptográficos comuns em execução em uma instância do EC2 com hsm2m.medium.

Dados de desempenho para hsm2m.medium

Operation	Cluster de dois HSM ¹	Cluster de três HSM ²	Cluster de seis HSM ³
Sinal RSA de 2048 bits	2000 operações/s	3000 operações/s	5000 operações/s

Operation	Cluster de dois HSM ¹	Cluster de três HSM ²	Cluster de seis HSM ³
Sinal EC P256	3000 operações/s	4500 operações/s	7000 operações/s

- [1] Um cluster de dois HSM com o aplicativo Java multiencadeado em execução em uma [instância do c4.large EC2](#) com um HSM na mesma AZ da instância do EC2.
- [2] Um cluster de três HSM com o aplicativo Java multiencadeado em execução em uma [instância do c4.large EC2](#) com um HSM na mesma AZ da instância do EC2.
- [3] Um cluster de seis HSM com o aplicativo Java multiencadeado em execução em uma [instância do c4.large EC2](#) com dois HSMs na mesma AZ da instância do EC2.

Controle de utilização do HSM

Quando sua carga de trabalho exceder a capacidade de HSM do cluster, você receberá mensagens de erro informando que HSMs está ocupado ou está limitado. Para obter detalhes sobre o que fazer quando isso acontece, consulte [Controle de utilização do HSM](#)

Segurança em AWS CloudHSM

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de data centers e arquiteturas de rede criados 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 segurança da nuvem e segurança na nuvem:

- **Segurança da nuvem** — AWS é responsável por proteger a infraestrutura que executa AWS os serviços no Nuvem AWS. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de [AWS](#) de . Para saber mais sobre os programas de conformidade que se aplicam AWS CloudHSM, consulte [AWS Serviços no escopo do programa de conformidade AWS](#) .
- **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.

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar AWS CloudHSM. Os tópicos a seguir mostram como configurar para atender AWS CloudHSM aos seus objetivos de segurança e conformidade. Você também aprende a usar outros serviços da AWS que ajudam você a monitorar e proteger seus AWS CloudHSM recursos.

Conteúdo

- [Controlar o acesso à API com políticas do IAM](#)
- [Proteção de dados em AWS CloudHSM](#)
- [Gerenciamento de identidade e acesso para AWS CloudHSM](#)
- [Validação de conformidade para AWS CloudHSM](#)
- [Resiliência em AWS CloudHSM](#)
- [Segurança da infraestrutura em AWS CloudHSM](#)
- [AWS CloudHSM e VPC endpoints](#)
- [Gerenciamento de atualizações em AWS CloudHSM](#)

Controlar o acesso à API com políticas do IAM

Atualize as políticas do IAM para IPv6

AWS CloudHSM os clientes usam políticas do IAM para controlar o acesso AWS CloudHSM APIs e impedir que qualquer endereço IP fora do intervalo configurado possa ser acessado AWS CloudHSM APIs.

O cloudhsmv2. *<region>*Endpoint de pilha dupla .api.aws onde estão AWS CloudHSM APIs hospedados suportes, além de. IPv6 IPv4

Clientes que precisam oferecer suporte a ambos IPv4 e IPv6 precisam atualizar suas políticas de filtragem de endereços IP para lidar com IPv6 endereços, caso contrário, isso afetará sua capacidade de se conectar a AWS CloudHSM mais IPv6.

Quem deve fazer a atualização?

Os clientes que usam endereçamento duplo com políticas que contêm aws:sourceip são afetados por essa atualização. O endereçamento duplo significa que a rede oferece suporte tanto para IPv4 quanto IPv6.

Se você estiver usando endereçamento duplo, deverá atualizar suas políticas do IAM que estão atualmente configuradas com endereços de IPv4 formato para incluir endereços de IPv6 formato.

Para obter ajuda com problemas de acesso, entre em contato com [Suporte](#).

Note

Os seguintes clientes não são afetados por essa atualização:

- Clientes que estão apenas em IPv4 redes.

O que é IPv6?

IPv6 é o padrão IP de próxima geração destinado a ser substituído eventualmente IPv4. A versão anterior, IPv4, usa um esquema de endereçamento de 32 bits para suportar 4,3 bilhões de dispositivos. IPv6 em vez disso, usa endereçamento de 128 bits para suportar aproximadamente 340 trilhões de trilhões de trilhões (ou 2 até a 128ª potência) de dispositivos.

Para obter mais detalhes, consulte a página da Web da [VPC IPv6](#).

```
2001:cdba:0000:0000:0000:0000:3257:9652
2001:cdba:0:0:0:0:3257:9652
2001:cdba::3257:965
```

Atualização de uma política do IAM para IPv6

Atualmente, as políticas do IAM são usadas para definir um intervalo permitido de endereços IP usando o filtro `aws:SourceIp`.

O endereçamento duplo suporta tanto o IPv6 tráfego IPv4 quanto o tráfego. Se sua rede usa endereçamento duplo, você deve atualizar todas as políticas do IAM usadas para filtragem de endereços IP para incluir intervalos de IPv6 endereços.

Por exemplo, a política abaixo identifica os intervalos IPv4 de endereços permitidos `192.0.2.0.*` e `203.0.113.0.*` no `Condition` elemento.

```
# https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_examples_aws_deny-ip.html
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "NotIpAddress": {
        "*aws:SourceIp": [
          "*192.0.2.0/24*",
          "*203.0.113.0/24*"
        ]
      },
      "Bool": {
        "aws:ViaAWSService": "false"
      }
    }
  }
}
```

Para atualizar essa política, altere o `Condition` elemento para incluir os intervalos de IPv6 endereços `2001:DB8:1234:5678::/64` `2001:cdba:3257:8593::/64` e.

Note

NÃO REMOVA os IPv4 endereços existentes porque eles são necessários para a compatibilidade com versões anteriores.

```
"Condition": {
    "NotIpAddress": {
        "*aws:SourceIp*": [
            "*192.0.2.0/24*", <<DO NOT REMOVE existing IPv4 address>>
            "*203.0.113.0/24*", <<DO NOT REMOVE existing IPv4 address>>
            "*2001:DB8:1234:5678::/64*", <<New IPv6 IP address>>
            "*2001:cdba:3257:8593::/64*" <<New IPv6 IP address>>
        ]
    },
    "Bool": {
        "aws:ViaAWSService": "false"
    }
}
```

Verifique o suporte do seu cliente IPv6

É recomendável que os clientes que usam o endpoint `cloudhsmv2.{region}.api.aws` verifiquem se conseguem se conectar a ele. As etapas a seguir descrevem como realizar a verificação.

Este exemplo usa Linux e curl versão 8.6.0 e usa os endpoints de [AWS CloudHSM serviço que IPv6 habilitaram endpoints](#) localizados no endpoint `api.aws`.

Note

Mude Região da AWS para a mesma região em que o cliente está localizado. Neste exemplo, usamos o endpoint `us-east-1`, ou seja, Leste dos EUA (Norte da Virgínia).

1. Determine se o endpoint é resolvido com um IPv6 endereço usando o comando a seguir `dig`.

```
dig +short AAAA cloudhsmv2.us-east-1.api.aws
2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
```

2. Determine se a rede do cliente pode fazer uma IPv6 conexão usando o `curl` comando a seguir. Um código de resposta 404 significa uma conexão bem-sucedida, enquanto um código de resposta 0 significa falha da conexão.

```
curl --ipv6 -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
%{response_code}\n" https://cloudhsmv2.us-east-1.api.aws

remote ip: 2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
response code: 404
```

Se um IP remoto foi identificado e o código de resposta não 0, uma conexão de rede foi estabelecida com sucesso com o endpoint usando IPv6. O IP remoto deve ser um IPv6 endereço porque o sistema operacional deve selecionar o protocolo válido para o cliente. Se o IP remoto não for um IPv6 endereço, use o comando a seguir para `curl` forçar o uso IPv4.

```
curl --ipv4 -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
%{response_code}\n" https://cloudhsmv2.us-east-1.api.aws

remote ip: 3.123.154.250
response code: 404
```

Se o IP remoto estiver em branco ou o código de resposta estiver 0, a rede do cliente ou o caminho da rede até o endpoint será somente IPv4 -. É possível verificar isso com o seguinte comando do `curl`:

```
curl -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
%{response_code}\n" https://cloudhsmv2.us-east-1.api.aws

remote ip: 3.123.154.250
response code: 404
```

Proteção de dados em AWS CloudHSM

O modelo de [responsabilidade AWS compartilhada modelo](#) se aplica à proteção de dados em AWS CloudHSM. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa 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 saber mais sobre a privacidade

de dados, consulte as [Data Privacy FAQ](#). Para saber mais sobre a proteção de dados na Europa, consulte a postagem do blog [AWS Shared Responsibility Model and RGPD](#) no Blog de segurança da AWS.

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com AWS IAM Identity Center 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 os 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 o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como [trabalhar com CloudTrail trilhas](#) no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente 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 AWS CloudHSM ou Serviços da AWS usa o console, a API ou 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.

Criptografia em repouso

Quando AWS CloudHSM faz um backup de um HSM, o HSM criptografa seus dados antes de enviá-los para. AWS CloudHSM Os dados são criptografados usando uma chave de criptografia exclusiva e temporária. Para obter mais informações, consulte [AWS CloudHSM backups de cluster](#).

Criptografia em trânsito

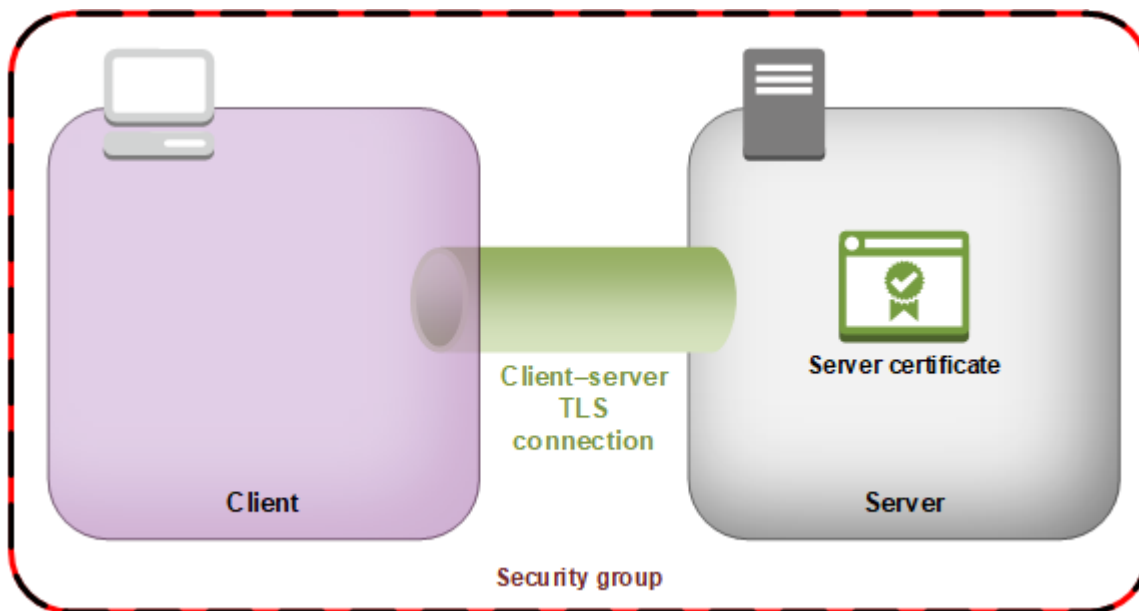
A comunicação entre o AWS CloudHSM cliente e o HSM em seu cluster é criptografada de ponta a ponta. Esta comunicação só pode ser descriptografada pelo seu cliente e seus HSMs. Para obter mais informações, consulte [End-to-end criptografia](#).

AWS CloudHSM end-to-endcriptografia de cliente

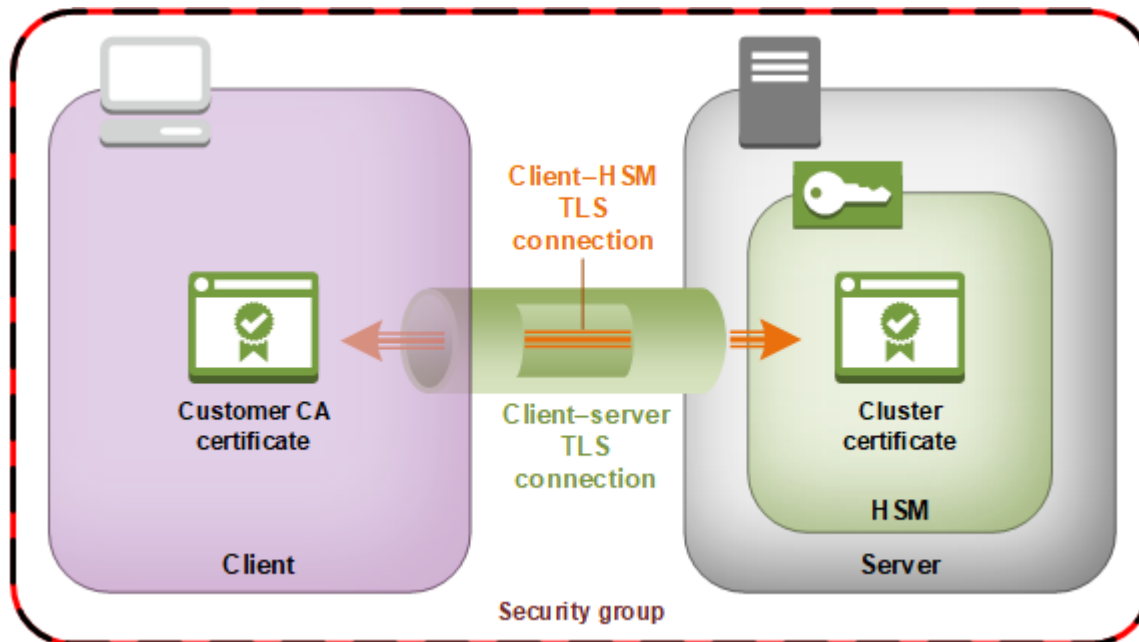
A comunicação entre a instância do cliente e a do HSMs seu cluster é criptografada de ponta a ponta. Somente seu cliente e você HSMs podem decifrar a comunicação.

O processo a seguir explica como o cliente estabelece comunicação end-to-end criptografada com um HSM.

1. O cliente estabelece uma conexão Transport Layer Security (TLS) com o servidor que hospeda o hardware de HSM. O security group do cluster permite que tráfego de entrada para o servidor apenas a partir de instâncias de clientes no security group. O cliente também verifica o certificado do servidor para garantir que é um servidor confiável.



2. A seguir, o cliente estabelece uma conexão criptografada com o hardware do HSM. O HSM tem o certificado de cluster que você assinou assinado com sua própria autoridade certificadora (CA), e o cliente tem o certificado raiz da CA. Antes de a conexão criptografada do HSM cliente ser estabelecida, o cliente verifica o certificado do cluster de HSM em relação ao seu certificado raiz. A conexão é estabelecida somente quando o cliente verificar com êxito se o HSM é confiável.



Segurança dos backups do cluster

Ao fazer AWS CloudHSM um backup do HSM, o HSM criptografa todos os seus dados antes de enviá-los para. AWS CloudHSM Os dados nunca saem do HSM em formato de texto simples. Além disso, os backups não podem ser descriptografados AWS porque AWS não tem acesso à chave usada para descriptografar os backups.

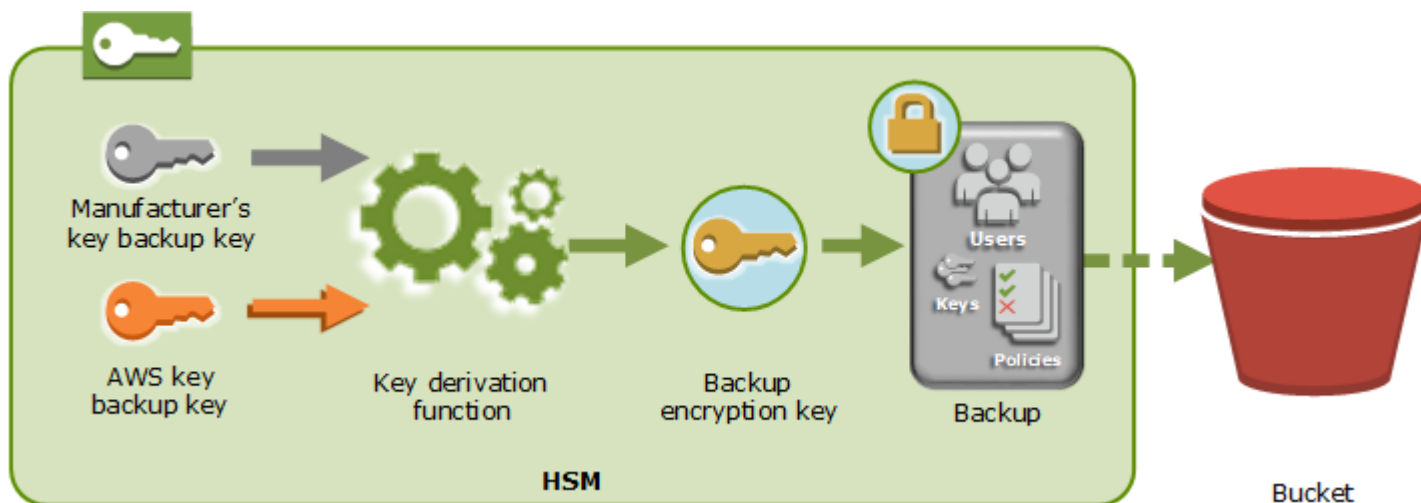
Para criptografar os dados, o HSM usa uma chave de criptografia temporária exclusiva como chave de backup temporária (EBK). O EBK é uma chave de criptografia AES de 256 bits gerada dentro do HSM ao AWS CloudHSM fazer um backup. O HSM gera a EBK e, em seguida, a utiliza para criptografar os dados do HSM com um método de encapsulamento de chave AES aprovado pelo FIPS e que está em conformidade com a [publicação especial do NIST 800-38F](#). Em seguida, o HSM fornece os dados criptografados para AWS CloudHSM. Esses dados criptografados incluem uma cópia criptografada da EBK.

Para criptografar a EBK, o HSM usa outra chave de criptografia conhecida como chave de backup persistente (PBK). A PBK também é uma chave de criptografia AES de 256 bits. Para gerar a PBK,

o HSM usa uma função de derivação de chaves (KDF) aprovada pelo FIPS no modo de contador compatível com a [publicação especial 800-108 do NIST](#). As entradas dessa KDF incluem o seguinte:

- Uma chave de backup da chave de fabricante (MKBK), permanentemente incorporada no hardware do HSM pelo fabricante.
- Uma AWS chave de backup de chave (AKBK), instalada com segurança no HSM quando inicialmente configurada pelo AWS CloudHSM

Os processos de criptografia estão resumidos na figura a seguir. A chave de criptografia de backup representa a chave de backup persistente (PBK) e a chave de backup efêmera (EBK).



AWS CloudHSM pode restaurar backups exclusivos AWS HSMs feitos pelo mesmo fabricante. Como cada backup contém todos os usuários, chaves e configuração do HSM original, o HSM restaurado contém as mesmas proteções e controles de acesso do original. Os dados restaurados substituem todos os outros dados que possam ter estado no HSM antes da restauração.

Um backup consiste apenas em dados criptografados. Antes de o serviço armazenar um backup no Amazon S3, o serviço criptografa o backup novamente usando AWS Key Management Service (AWS KMS).

Gerenciamento de identidade e acesso para AWS CloudHSM

A AWS usa credenciais de segurança para identificar você e conceder acesso aos recursos da AWS. Você pode usar os recursos do AWS Identity and Access Management (IAM) para permitir que outros usuários, serviços e aplicativos usem seus recursos da AWS de forma total ou limitada. É possível fazer isso sem compartilhar as credenciais de segurança.

Por padrão, os usuários do IAM não têm permissão para criar, visualizar ou modificar os recursos da AWS. Para permitir que um usuário do IAM acesse recursos, como um load balancer, e execute tarefas, você:

1. Criar uma política do IAM que conceda permissão ao usuário do IAM para usar os recursos específicos e ações de API de que ele precisa.
2. Anexar a política ao usuário do IAM ou ao grupo ao qual o usuário do IAM pertence.

Ao anexar uma política a um usuário ou grupo de usuários, isso concede ou nega aos usuários permissão para realizar as tarefas especificadas nos atributos especificados.

Por exemplo, é possível usar o IAM para criar usuários e grupos na conta da AWS. Um usuário do IAM pode ser uma pessoa, um sistema ou um aplicativo. Em seguida, você concede permissões aos usuários e grupos para executar ações específicas nos recursos especificados usando uma política do IAM;.

Conceder permissões usando políticas do IAM;

Ao anexar uma política a um usuário ou grupo de usuários, isso concede ou nega aos usuários permissão para realizar as tarefas especificadas nos atributos especificados.

A política do IAM é um documento JSON que consiste em uma ou mais instruções.

O exemplo de política do IAM a seguir permite que os usuários descrevam AWS CloudHSM backups no Leste dos EUA (Norte da Virgínia). Somente as solicitações descritas provenientes do Leste dos EUA (Norte da Virgínia) são permitidas.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloudhsm:DescribeBackups",
      "Resource": "arn:aws:cloudhsm:us-east-1:123456789012:backup/*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "us-east-1"
        }
      }
    }
  ]
}
```

```
}  
}  
]  
}
```

- **Efeito:** o efeito pode ser Allow ou Deny. Por padrão, os usuários do IAM não têm permissão para usar recursos e ações da API. Por isso, todas as solicitações são negadas. Uma permissão explícita substitui o padrão. Uma negar explícito substitui todas as permissões.
- **Ação:** a ação é a ação de API específica para a qual você está concedendo ou negando permissão. Para obter mais informações sobre como especificar a ação, consulte [Ações de API para AWS CloudHSM](#).
- **Recurso** — O recurso afetado pela ação. AWS CloudHSM não oferece suporte a permissões em nível de recurso. Você deve usar o caractere curinga * para especificar todos os AWS CloudHSM recursos.
- **Condição:** fica a seu critério usar as condições para controlar quando a política estará em vigor. Para obter mais informações, consulte [Chaves de condição para AWS CloudHSM](#).

Para saber mais, consulte o [Manual do usuário do IAM](#).

Ações de API para AWS CloudHSM

No elemento Ação da sua declaração de política do IAM, você pode especificar qualquer ação de API que AWS CloudHSM ofereça. É necessário prefixar o nome da ação com a string em minúsculas `cloudhsm:`, conforme mostrado no exemplo a seguir.

```
"Action": "cloudhsm:DescribeClusters"
```

Para especificar várias ações em uma única instrução, coloque-as entre colchetes e separe-as com vírgula, como mostrado no exemplo a seguir.

```
"Action": [  
    "cloudhsm:DescribeClusters",  
    "cloudhsm:DescribeHsm"  
]
```

Você também pode especificar várias ações usando o caractere curinga *. O exemplo a seguir especifica todos os nomes de ações de API AWS CloudHSM que começam com `List`.


```
"Action": "cloudhsm:List*"
```

Para especificar todas as ações da API para AWS CloudHSM, use o caractere curinga *, conforme mostrado no exemplo a seguir.

```
"Action": "cloudhsm:*"
```

Para ver a lista de ações de API para AWS CloudHSM, consulte [AWS CloudHSM Ações](#).

Chaves de condição para AWS CloudHSM

Ao criar uma política, você pode especificar as condições que controlam quando a política está em vigor. Cada condição contém um ou mais pares de chave-valor. Há chaves de condição global e chaves de condição específicas do serviço.

AWS CloudHSM não tem chaves de contexto específicas do serviço.

Para obter mais informações sobre chaves de condição global, consulte [Chaves de contexto de condição global da AWS](#) no Guia de usuário da IAM.

Políticas predefinidas gerenciadas pela AWS para AWS CloudHSM

As políticas gerenciadas criadas pela AWS concedem as permissões necessárias para casos de uso comuns. É possível anexar essas políticas aos usuários do IAM de acordo com o acesso de que eles precisam no AWS CloudHSM :

- **AWSCloudHSMFullAcesso** — Concede o acesso total necessário para usar os AWS CloudHSM recursos.
- **AWSCloudHSMReadOnlyAccess**— Concede acesso somente para AWS CloudHSM leitura aos recursos.

Políticas gerenciadas pelo cliente para AWS CloudHSM

Recomendamos que você crie um grupo de administradores do IAM AWS CloudHSM que contenha somente as permissões necessárias para execução AWS CloudHSM. Anexe a política com as permissões apropriadas a este grupo. Adicione usuários do IAM; ao grupo, conforme necessário. Cada usuário que você adicionar herda a política do grupo de administradores.

Além disso, recomendamos criar grupos de usuários adicionais de acordo com as permissões que seus usuários precisam. Isso garante que somente usuários confiáveis tenham acesso a ações críticas da API. Por exemplo, você pode criar um grupo de usuários que você usa para conceder acesso somente de leitura aos clusters e HSMs. Como esse grupo não permite que um usuário exclua clusters ou HSMs, um usuário não confiável não pode afetar a disponibilidade de uma carga de trabalho de produção.

À medida que novos recursos AWS CloudHSM de gerenciamento são adicionados ao longo do tempo, você pode garantir que somente usuários confiáveis tenham acesso imediato. Ao atribuir permissões limitadas a políticas na criação, você poderá atribuir manualmente novas permissões de atributo posteriormente.

A seguir estão exemplos de políticas para AWS CloudHSM. Para obter informações sobre como criar uma política e anexá-la a um grupo de usuários do IAM, consulte [Criar políticas na guia JSON](#) no Guia usuário do IAM.

Exemplos

- [Permissões somente leitura](#)
- [Permissões de usuário avançado](#)
- [Permissões de administrador](#)

Exemplo: permissões somente leitura

Esta política permite o acesso às ações da API `DescribeClusters` e `DescribeBackups`. Ela também inclui permissões adicionais para ações da API específicas do Amazon EC2 API. Ele não permite que o usuário exclua clusters ou HSMs.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudhsm:DescribeClusters",
        "cloudhsm:DescribeBackups",
        "cloudhsm:ListTags"
      ]
    }
  ]
}
```

```

    "Resource": "*"
  }
}

```

Example Exemplo: permissões de usuário avançado

Essa política permite o acesso a um subconjunto das ações da AWS CloudHSM API. Ela também inclui permissões adicionais para ações específicas do Amazon EC2. Ele não permite que o usuário exclua clusters ou HSMs. Você deve incluir a `iam:CreateServiceLinkedRole` ação para permitir AWS CloudHSM a criação automática da função vinculada ao serviço `AWSServiceRoleForCloudHSM` em sua conta. Essa função permite AWS CloudHSM registrar eventos. Para obter mais informações, consulte [Funções vinculadas a serviços para AWS CloudHSM](#).

Note

Para permissões específicas por API, consulte a tabela de [Ações, recursos e chaves de condição do AWS CloudHSM](#) na Referência de autorização do serviço.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "cloudhsm:DescribeClusters",
      "cloudhsm:DescribeBackups",
      "cloudhsm:CreateCluster",
      "cloudhsm:CreateHsm",
      "cloudhsm:RestoreBackup",
      "cloudhsm:CopyBackupToRegion",
      "cloudhsm:InitializeCluster",
      "cloudhsm:ListTags",
      "cloudhsm:TagResource",
      "cloudhsm:UntagResource",
      "ec2:CreateNetworkInterface",
      "ec2:DescribeNetworkInterfaces",

```

```

        "ec2:DescribeNetworkInterfaceAttribute",
        "ec2:DetachNetworkInterface",
        "ec2>DeleteNetworkInterface",
        "ec2:CreateSecurityGroup",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:DescribeSecurityGroups",
        "ec2>DeleteSecurityGroup",
        "ec2:CreateTags",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets",
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "*"
}
}

```

Example Exemplo: permissões de administrador

Essa política permite acesso a todas as ações AWS CloudHSM da API, incluindo as ações para excluir HSMs e clusters. Ela também inclui permissões adicionais para ações específicas do Amazon EC2. Você deve incluir a `iam:CreateServiceLinkedRole` ação para permitir AWS CloudHSM a criação automática da função vinculada ao serviço `AWSServiceRoleForCloudHSM` em sua conta. Essa função permite AWS CloudHSM registrar eventos. Para obter mais informações, consulte [Funções vinculadas a serviços para AWS CloudHSM](#).

JSON

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "cloudhsm:*",
      "ec2:CreateNetworkInterface",
      "ec2:DescribeNetworkInterfaces",
      "ec2:DescribeNetworkInterfaceAttribute",
      "ec2:DetachNetworkInterface",
      "ec2>DeleteNetworkInterface",
      "ec2:CreateSecurityGroup",

```

```

        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:DescribeSecurityGroups",
        "ec2>DeleteSecurityGroup",
        "ec2:CreateTags",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets",
        "iam:CreateServiceLinkedRole"
    ],
    "Resource": "*"
}
}

```

Funções vinculadas a serviços para AWS CloudHSM

A política do IAM que você criou anteriormente [Políticas gerenciadas pelo cliente para AWS CloudHSM](#) inclui a `iam:CreateServiceLinkedRole` ação. AWS CloudHSM define uma [função vinculada ao serviço chamada `AWSService RoleForCloud HSM`](#). A função é predefinida AWS CloudHSM e inclui permissões que AWS CloudHSM exigem chamar outros AWS serviços em seu nome. A função facilita a configuração de um serviço, pois você não precisa adicionar manualmente as permissões das políticas de função e de confiança.

A política de função permite AWS CloudHSM criar grupos e fluxos de log do Amazon CloudWatch Logs e gravar eventos de log em seu nome. Você pode visualizar essa política abaixo e no console do IAM.

JSON

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogStreams"
      ]
    }
  ]
}

```

```

    ],
    "Resource": [
        "arn:aws:logs:*:*:*"
    ]
}
]
}

```

A política de confiança para a função de AWSServiceRoleForCloudHSM permite que você assuma AWS CloudHSM a função.

JSON

```

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

```

Criar uma função vinculada a serviços (automática)

AWS CloudHSM cria a função AWSServiceRoleForCloudHSM ao criar um cluster se você incluir a `iam:CreateServiceLinkedRole` ação nas permissões definidas ao criar o grupo de AWS CloudHSM administradores. Consulte [Políticas gerenciadas pelo cliente para AWS CloudHSM](#).

Se você já tem um ou mais clusters e quer apenas adicionar a função

AWSServiceRoleForCloudHSM, você pode usar o console, o comando [create-cluster](#) ou a operação da [CreateCluster](#) API para criar um cluster. Em seguida, use o console, o comando [delete-cluster](#) ou a operação da [DeleteCluster](#) API para excluí-lo. Criar outro cluster cria a função vinculada ao serviço e a aplica a todos os clusters em sua conta. Como alternativa, você pode criar a função manualmente. Consulte a seção a seguir para ter mais informações.

 Note

Você não precisa executar todas as etapas descritas em [Começando com AWS CloudHSM](#) para criar um cluster se estiver apenas criando-o para adicionar a função AWSServiceRoleForCloudHSM.

Criar uma função vinculada a serviços (manual)

Você pode usar o console do IAM ou a API para criar a função do AWSServiceRoleForCloudHSM. AWS CLI Para obter mais informações, consulte [Criar uma função vinculada ao serviço](#) no Guia do usuário do IAM.

Edição da função vinculada ao serviço

AWS CloudHSM não permite que você edite a função do AWSServiceRoleForCloudHSM. Por exemplo, depois que a função é criada, você não pode alterar o seu nome, pois várias entidades podem fazer referência à função pelo nome. Além disso, você não pode alterar a política da função. É possível, no entanto, usar o IAM para editar a descrição da função. Para obter mais informações, consulte [Editar uma função vinculada ao serviço](#) no Guia do usuário do IAM.

Excluindo uma função vinculada ao serviço

Não é possível excluir uma função vinculada ao serviço enquanto ainda houver um cluster aplicado a ela. Para excluir a função, você deve primeiro excluir cada HSM do cluster e, em seguida, excluir o cluster. Todos os clusters de sua conta devem ser excluídos. Em seguida, você pode usar o console do IAM ou a API para excluir a função. AWS CLI Para obter mais informações sobre a exclusão de um cluster, consulte [Excluindo um cluster AWS CloudHSM](#). Para saber mais, consulte [Excluir um perfil vinculado ao serviço](#) no Guia do usuário do IAM.

Validação de conformidade para AWS CloudHSM

Para clusters no modo FIPS, AWS CloudHSM fornece aprovação FIPS HSMs que atendem aos requisitos de PCI-PIN, PCI-3DS e conformidade. SOC2 AWS CloudHSM também oferece aos clientes a opção de escolher clusters que não estejam no modo FIPS. Para obter detalhes sobre quais requisitos de certificação e conformidade se aplicam a cada um deles, consulte [AWS CloudHSM modos de cluster](#).

Contar com um HSM validado pelo FIPS pode ajudá-lo a atender aos requisitos de conformidade corporativa, contratual e regulatória para segurança de dados na nuvem. AWS

Conformidade com o FIPS 140-2

A publicação 140-2 do Federal Information Processing Standard (FIPS - Padrão de processamento de informações federal) é um padrão de segurança do governo dos Estados Unidos que especifica os requisitos de segurança para módulos de criptografia que protegem informações confidenciais. [O tipo de instância AWS CloudHSM hsm1.medium tem certificação FIPS 140-2 nível 3 \(Certificado #4218\)](#). Em 4 de janeiro de 2026, o certificado para hsm1.medium passará para a lista histórica. Recomendamos que os clientes migrem para o hsm2m.medium, que tem a certificação FIPS 140-3 ([Certificado nº 4703](#)). Para obter mais informações, consulte [Validação FIPS para hardware](#).

Conformidade com o FIPS 140-3

A publicação 140-3 do Federal Information Processing Standard (FIPS - Padrão de processamento de informações federal) é um padrão de segurança do governo dos Estados Unidos que especifica os requisitos de segurança para módulos de criptografia que protegem informações confidenciais. [O tipo hsm2m.medium HSMs fornecido pela AWS CloudHSM é certificado FIPS 140-3 de nível 3 \(Certificado #4703\)](#). Para obter mais informações, consulte [Validação FIPS para hardware](#).

[Compatibilidade com PCI DSS](#)

O PCI DSS (Payment Card Industry Data Security Standard) é um padrão de segurança da informação exclusivo administrado pelo [PCI Security Standards Council](#). Os HSMs fornecidos por AWS CloudHSM estão em conformidade com o PCI DSS.

[Compatibilidade com PCI PIN](#)

O PCI PIN fornece requisitos de segurança e padrões de avaliação para transmissão, processamento e gerenciamento de dados de número de identificação pessoal (PIN), informações que são usadas para transações em ATMs terminais point-of-sale (POS). O hsm1.medium e o hsm2m.medium fornecidos pela são compatíveis com HSMs PCI PIN. AWS CloudHSM Para obter mais informações, consulte o artigo [O AWS CloudHSM agora tem certificação PCI PIN](#).

Compatibilidade com PCI-3DS

O PCI 3DS (ou Three Domain Secure, 3-D Secure) fornece segurança de dados para pagamentos de comércio eletrônico seguros EMV 3D. O PCI 3DS fornece outra camada de

segurança para compras on-line. O hsm1.medium e o hsm2m.medium fornecidos pela são compatíveis com HSMs PCI-3DS. AWS CloudHSM

SOC2

SOC2 é uma estrutura para ajudar organizações de serviços a demonstrar seus controles de segurança na nuvem e no data center. O AWS CloudHSM SOC2 implementou controles em áreas críticas para aderir aos princípios de serviços confiáveis. Para obter mais informações, consulte [a FAQs página SOC da AWS](#).

AWS CloudHSM Conformidade com PCI-PIN FAQs

O PCI PIN fornece requisitos de segurança e padrões de avaliação para transmissão, processamento e gerenciamento de dados de número de identificação pessoal (PIN), informações que são usadas para transações em ATMs terminais point-of-sale (POS).

O Atestado de Conformidade (AOC) e o resumo da responsabilidade do PCI-PIN estão disponíveis para os clientes por meio do AWS Artifact, um portal de autoatendimento para acesso sob demanda aos relatórios de conformidade da AWS. Para obter mais informações, faça login no [AWS Artifact no Console de Gerenciamento da AWS](#) ou saiba mais em [Como começar com AWS Artifact](#).

FAQs

P: O que é o resumo do atestado de conformidade e responsabilidade?

O Atestado de Conformidade (AOC) é produzido por um avaliador qualificado de PIN (QPA), atestando que AWS CloudHSM atende aos controles aplicáveis no padrão PCI-PIN. A matriz resumida de responsabilidades descreve os controles que são as respectivas responsabilidades AWS CloudHSM e de seus clientes.

P: Como faço para obter o AWS CloudHSM Atestado de Conformidade?

O Atestado de Conformidade (AOC) do PCI-PIN está disponível para os clientes por meio do AWS Artifact, um portal de autoatendimento para acesso sob demanda aos relatórios de conformidade da AWS. Para obter mais informações, faça login no [AWS Artifact no Console de Gerenciamento da AWS](#) ou saiba mais em [Como começar com AWS Artifact](#).

P: Como posso saber por quais controles PCI PIN sou responsável?

Para obter informações detalhadas, consulte o “Resumo da Responsabilidade do PIN do AWS CloudHSM PCI” do Pacote de Conformidade de PIN do PCI da AWS, disponível para clientes por

meio do AWS Artifact, um portal de autoatendimento para acesso sob demanda aos relatórios de conformidade da AWS. Para obter mais informações, faça login no [AWS Artifact no Console de Gerenciamento da AWS](#) ou saiba mais em [Como começar com AWS Artifact](#).

P: Como AWS CloudHSM cliente, posso confiar no Atestado de Conformidade (AOC) do PCI-PIN?

Os clientes devem gerenciar sua própria conformidade com o PCI-PIN. Você deve passar por um processo formal de atestação de PCI-PIN por meio de um avaliador qualificado de PIN (QPA) para verificar se sua workload de pagamento satisfaz todos os controles/requisitos de PCI-PIN. No entanto, para os controles pelos quais a AWS é responsável, seu QPA pode confiar no AWS CloudHSM Atestado de Conformidade (AOC) sem testes adicionais.

P: É AWS CloudHSM responsável pelos requisitos de PCI-PIN relacionados ao ciclo de vida do gerenciamento de chaves?

AWS CloudHSM é responsável pelo ciclo de vida do dispositivo físico do. HSMs Os clientes são responsáveis pelos principais requisitos do ciclo de vida do gerenciamento no padrão PCI-PIN.

P: Quais AWS CloudHSM controles são compatíveis com PCI-PIN?

O AOC resume os AWS CloudHSM controles que são avaliados pelo QPA. O resumo da responsabilidade do PCI-PIN está disponível para os clientes por meio do AWS Artifact, um portal de autoatendimento para acesso sob demanda aos relatórios de conformidade da AWS.

P: AWS CloudHSM Suporta funções de pagamento, como tradução de PIN e DUKPT?

Não, AWS CloudHSM fornece um propósito geral HSMs. Com o tempo, podemos fornecer funções de pagamento. Embora o serviço não execute funções de pagamento diretamente, o atestado de conformidade com o AWS CloudHSM PCI PIN permite que os clientes obtenham sua própria conformidade com o PCI para seus serviços em execução. AWS CloudHSM Se você estiver interessado em usar os serviços de criptografia de pagamento da AWS para sua workload, consulte o blog [“Mova o processamento de pagamentos para a nuvem com a criptografia de pagamentos da AWS”](#).

Notificações de suspensão

De tempos em tempos, AWS CloudHSM pode descontinuar a funcionalidade para permanecer em conformidade com os requisitos do FIPS 140, PCI-DSS, PCI-PIN, PCI-3DS ou devido ao hardware. SOC2 end-of-support Esta página lista as alterações que se aplicam atualmente.

HSM1 Depreciação

O tipo de instância AWS CloudHSM `hsm1.medium` chegará ao fim do suporte em 31 de março de 2026. Para garantir a continuidade do serviço, estamos introduzindo as seguintes mudanças:

- A partir de abril de 2025, não será possível criar novos clusters `hsm1.medium`.
- A partir de janeiro de 2026, começaremos a migrar automaticamente os clusters `hsm1.medium` existentes para o novo tipo de instância `hsm2m.medium`.

O tipo de instância `hsm2m.medium` é compatível com seu tipo de AWS CloudHSM instância atual e oferece desempenho aprimorado. Para evitar interrupções em suas aplicações, faça a atualização para a versão mais recente do SDK do cliente. Para obter instruções de atualização, consulte [???](#).

Você tem duas opções para migração:

1. Opte por uma migração gerenciada pelo CloudHSM quando estiver tudo pronto. Para obter mais informações, [???](#).
2. Crie um novo cluster `hsm2m.medium` de um backup do cluster `hsm1` e redirecione a aplicação para o novo cluster. Recomendamos usar uma estratégia blue/green de implantação para essa abordagem. Para obter mais informações, consulte [???](#).

Conformidade com o FIPS 140: suspensão do mecanismo de 2024

O Instituto Nacional de Padrões e Tecnologia (NIST) ¹informa que o suporte à criptografia Triple DES (DESede, 3DES, DES3) e ao empacotamento e desempacotamento de chaves RSA com preenchimento PKCS #1 v1.5 não é permitido após 31 de dezembro de 2023. Portanto, o suporte para eles termina em 1.º de janeiro de 2024 em nossos clusters no modo Federal Information Processing Standard (FIPS – Padrão federal de processamento de informações). Support para eles permanece para clusters em FIPs modo não-.

Essa orientação se aplica às seguintes operações criptográficas:

- Geração tripla de chaves DES
 - CKM_DES3_KEY_GEN para a Biblioteca PKCS #11
 - DESede Keygen para o provedor JCE
 - genSymKey com `-t=21` para o KMU
- Criptografia com chaves DES triplas (observação: operações de descriptografia são permitidas)

- Para a biblioteca PKCS #11: criptografe CKM_DES3_CBC, criptografe CKM_DES3_CBC_PAD e , e criptografe CKM_DES3_ECB
- Para o provedor JCE: criptografe DESede/CBC/PKCS5Padding, criptografe DESede/CBC/NoPadding, criptografe DESede/ECB/Padding e criptografe DESede/ECB/NoPadding
- Encapsulamento, desencapsulamento, criptografia e descriptografia de chaves RSA com o preenchimento PKCS #1 v1.5
- CKM_RSA_PKCS encapsulamento, desencapsulamento, criptografia e descriptografia para o SDK PKCS #11
- RSA/ECB/PKCS1Padding encapsulamento, desencapsulamento, criptografia e descriptografia para o SDK JCE
- wrapKey e unwrapKey com -m 12 para o KMU (a nota 12 é o valor do mecanismo RSA_PKCS)

[1] Para obter detalhes sobre essa alteração, consulte a Tabela 1 e a Tabela 5 em [Transição do uso de algoritmos criptográficos e cumprimentos de chave](#).

Resiliência em AWS CloudHSM

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. AWS As regiões fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. 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](#). Para obter mais informações sobre atributos do AWS CloudHSM para oferecer suporte à resiliência, consulte [AWS CloudHSM alta disponibilidade e balanceamento de carga do cluster](#).

Segurança da infraestrutura em AWS CloudHSM

Como serviço gerenciado, AWS CloudHSM 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 CloudHSM pela rede. 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 rede

Uma nuvem privada virtual (VPC) é uma rede virtual na área isolada logicamente na Nuvem AWS. É possível criar um cluster em uma sub-rede privada em sua VPC. Você pode criar sub-redes privadas ao criar uma VPC. Para obter mais informações, consulte [Crie uma nuvem privada virtual \(VPC\) para AWS CloudHSM](#).

Ao criar um HSM, AWS CloudHSM coloque uma interface de rede elástica (ENI) em sua sub-rede para que você possa interagir com sua. HSMs Para obter mais informações, consulte [AWS CloudHSM arquitetura de cluster](#).

AWS CloudHSM cria um grupo de segurança que permite a comunicação de entrada e saída entre HSMs em seu cluster. Você pode usar esse grupo de segurança para permitir que suas instâncias do EC2 se comuniquem com as do HSMs seu cluster. Para obter mais informações, consulte [Configure os grupos de segurança da instância cliente Amazon EC2 para AWS CloudHSM](#).

Autorização dos usuários

Com AWS CloudHSM, as operações realizadas no HSM exigem as credenciais de um usuário autenticado do HSM. Para obter mais informações, consulte [the section called “Tipos de usuário”](#).

AWS CloudHSM e VPC endpoints

Você pode estabelecer uma conexão privada entre sua VPC e criar uma AWS CloudHSM interface VPC endpoint. Os endpoints de interface são alimentados por [AWS PrivateLink](#) uma tecnologia que permite que você acesse de forma privada AWS CloudHSM APIs sem um gateway de internet, dispositivo NAT, conexão VPN ou conexão do AWS Direct Connect. As instâncias em sua VPC não precisam de endereços IP públicos para se comunicar. AWS CloudHSM APIs O tráfego de rede entre a VPC e o AWS CloudHSM não deixa a rede da Amazon.

Cada endpoint de interface é representado por uma ou mais [Interfaces de Rede Elástica](#) nas sub-redes.

Para obter mais informações, consulte [Interface VPC endpoints \(AWS PrivateLink\)](#) no Guia do usuário da Amazon VPC.

Considerações sobre AWS CloudHSM VPC endpoints

Antes de configurar uma interface para o VPC endpoint AWS CloudHSM, certifique-se de revisar as [propriedades e limitações do endpoint da interface no](#) Guia do usuário do Amazon VPC.

- AWS CloudHSM suporta fazer chamadas para todas as suas ações de API a partir de sua VPC.

Criação de uma interface VPC endpoint para AWS CloudHSM

Você pode criar um VPC endpoint para o AWS CloudHSM serviço usando o console Amazon VPC ou o (). AWS Command Line Interface AWS CLI Para obter mais informações, consulte [Criar um endpoint de interface](#) no Guia do usuário da Amazon VPC.

Para criar um VPC endpoint para AWS CloudHSM, use o seguinte nome de serviço:

```
com.amazonaws.<region>.cloudhsmv2
```

Por exemplo, na Região Oeste dos EUA (Oregon) (us-west-2), o nome do serviço seria:

```
com.amazonaws.us-west-2.cloudhsmv2
```

Para facilitar o uso do endpoint da VPC, é possível habilitar um [nome de host DNS privado](#) para o endpoint da VPC. Se você selecionar a opção Habilitar nome DNS privado, os nomes de host AWS CloudHSM DNS padrão (`https://cloudhsmv2.<region>.amazonaws.com` e `https://cloudhsmv2.<region>.api.aws`) serão resolvidos para seu VPC endpoint.

Essa opção facilita usar o endpoint da VPC. O AWS SDKs e AWS CLI usa o nome de host AWS CloudHSM DNS padrão por padrão, então você não precisa especificar a URL do VPC endpoint em aplicativos e comandos.

Para mais informações, consulte [Acessar um serviço por um endpoint de interface](#) no Guia do usuário da Amazon VPC.

Criação de uma política de VPC endpoint para AWS CloudHSM

É possível anexar uma política de endpoint ao endpoint da VPC que controla o acesso ao AWS CloudHSM. Essa política especifica as seguintes informações:

- A entidade principal que pode realizar ações.

- As ações que podem ser realizadas.
- Os recursos aos quais as ações podem ser aplicadas.

Para mais informações, consulte [Controlar o acesso a serviços com VPC endpoints](#) no Guia do usuário da Amazon VPC.

Exemplo: política de VPC endpoint para ações AWS CloudHSM

Veja a seguir um exemplo de uma política de endpoint para AWS CloudHSM. Quando anexada a um endpoint, essa política concede acesso às AWS CloudHSM ações listadas para todos os diretores em todos os recursos. Consulte [Gerenciamento de identidade e acesso para AWS CloudHSM](#) para ver outras AWS CloudHSM ações e as permissões correspondentes do IAM.

```
{
  "Statement": [
    {
      "Principal": "*",
      "Effect": "Allow",
      "Action": [
        "<cloudhsm>:<DescribeBackups>",
        "<cloudhsm>:<DescribeClusters>",
        "<cloudhsm>:<ListTags>",
      ],
      "Resource": "*"
    }
  ]
}
```

Gerenciamento de atualizações em AWS CloudHSM

A AWS gerencia o firmware. O firmware é mantido por um terceiro e deve ser avaliado pela NIST para comprovar a conformidade com o FIPS 140-2 nível 3 ou FIPS 140-3 nível 3, dependendo do tipo de hsm. Apenas o firmware criptograficamente assinado pela chave do FIPS (à qual a AWS não tem acesso) poderá ser instalado.

Solução de problemas AWS CloudHSM

Se você encontrar problemas com AWS CloudHSM, os tópicos a seguir podem ajudá-lo a resolvê-los.

Tópicos

- [AWS CloudHSM problemas conhecidos](#)
- [AWS CloudHSM Falhas de sincronização de teclas do Client SDK 3](#)
- [AWS CloudHSM O SDK 3 do cliente verifica o desempenho do HSM com a ferramenta pkpspeed](#)
- [AWS CloudHSM O usuário do SDK 5 do cliente contém valores inconsistentes](#)
- [AWS CloudHSM Falhas de replicação do usuário do Client SDK 5](#)
- [AWS CloudHSM Falhas de replicação de 5 chaves do SDK do cliente](#)
- [AWS CloudHSM erro visto durante a verificação da disponibilidade da chave](#)
- [AWS CloudHSM extraindo chaves usando o JCE](#)
- [Controle de utilização do HSM](#)
- [Mantenha os usuários do HSM sincronizados HSMs em todo o cluster AWS CloudHSM](#)
- [Conexão perdida com o AWS CloudHSM cluster](#)
- [Registros AWS CloudHSM de auditoria ausentes CloudWatch](#)
- [Personalizado IVs com comprimento não compatível para embalagem de chave AES AWS CloudHSM](#)
- [Resolvendo falhas na criação de AWS CloudHSM clusters](#)
- [Recuperando registros de configuração AWS CloudHSM do cliente](#)

AWS CloudHSM problemas conhecidos

AWS CloudHSM tem os seguintes problemas conhecidos. Escolha uma área de tópico para saber mais.

Tópicos

- [Problemas conhecidos para todas as instâncias do HSM](#)
- [Problemas conhecidos para instâncias AWS CloudHSM hsm1.medium](#)

- [Problemas conhecidos para instâncias AWS CloudHSM hsm2m.medium](#)
- [Problemas conhecidos da biblioteca PKCS #11 para AWS CloudHSM](#)
- [Problemas conhecidos do JCE SDK para AWS CloudHSM](#)
- [Problemas conhecidos do OpenSSL Dynamic Engine para AWS CloudHSM](#)
- [Problemas conhecidos do Key Storage Provider \(KSP\) para AWS CloudHSM](#)
- [Problemas conhecidos do OpenSSL Provider para AWS CloudHSM](#)
- [Problemas conhecidos para instâncias do Amazon EC2 executando o Amazon Linux 2 com AWS CloudHSM](#)
- [Problemas conhecidos na integração de aplicativos de terceiros com AWS CloudHSM](#)
- [Problemas conhecidos de modificação AWS CloudHSM do cluster](#)
- [Problemas conhecidos de falha de operação usando a versão 5.12.0 do AWS CloudHSM cliente no hsm2.medium](#)

Problemas conhecidos para todas as instâncias do HSM

Os problemas a seguir afetam todos os AWS CloudHSM usuários, independentemente de eles usarem a ferramenta de linha de comando `key_mgmt_util`, o SDK PKCS #11, o JCE SDK ou o OpenSSL SDK.

Tópicos

- [Problema: o encapsulamento de chave AES usa o preenchimento de PKCS #5 em vez de fornecer uma implementação compatível com os padrões do encapsulamento de chave com preenchimento de zeros](#)
- [Problema: o daemon do cliente requer pelo menos um endereço IP válido em seu arquivo de configuração para se conectar com sucesso ao cluster](#)
- [Problema: havia um limite máximo de 16 KB em dados que podem ser criptografados e assinados AWS CloudHSM usando o SDK do cliente 3](#)
- [Problema: as chaves importadas não podiam ser especificadas como não exportáveis](#)
- [Problema: o mecanismo padrão para o `WrapKey` e os `unWrapKey` comandos no `key\_mgmt\_util` foi removido](#)
- [Problema: se você tiver um único HSM em seu cluster, o failover do HSM não funcionará corretamente](#)

- [Problema: se você exceder a capacidade principal do HSMs em seu cluster em um curto período de tempo, o cliente entrará em um estado de erro não tratado](#)
- [Problema: operações de resumo com chaves HMAC de tamanho maior que 800 bytes não são compatíveis.](#)
- [Problema: a ferramenta client_info, distribuída com o Client SDK 3, exclui o conteúdo do caminho especificado pelo argumento de saída opcional](#)
- [Problema: você recebe um erro ao executar a ferramenta de configuração do SDK 5 usando o argumento --cluster-id em ambientes em contêineres](#)
- [Problema: você recebe o erro "Falha ao criar a cert/key partir do arquivo pfx fornecido. Erro: NotPkcs 8"](#)
- [Problema: falha na assinatura do ECDSA com o erro "mecanismo inválido" a partir do SDK 5.16](#)
- [Problema: as operações de assinatura com dados pré-hash não apagam adequadamente os tokens de sessão no modo interativo](#)
- [Problema: o certificado de cliente padrão da biblioteca de cliente do CloudHSM expira em 31 de janeiro de 2026](#)

Problema: o encapsulamento de chave AES usa o preenchimento de PKCS #5 em vez de fornecer uma implementação compatível com os padrões do encapsulamento de chave com preenchimento de zeros

Além disso, o encapsulamento de chaves sem preenchimento e com preenchimento de zeros não é compatível.

- Impacto: não haverá impacto se você embrulhar e desembulhar usando esse algoritmo interno AWS CloudHSM. No entanto, as chaves embaladas com AWS CloudHSM não podem ser desempacotadas em outro software HSMs ou em outro software que espere conformidade com a especificação sem preenchimento. Isso ocorre porque até oito bytes de dados de preenchimento podem ser adicionados ao final dos dados da chave durante um desencapsulamento compatível com os padrões. As chaves agrupadas externamente não podem ser desagrupadas adequadamente em uma AWS CloudHSM instância.
- Solução alternativa: para desencapsular externamente uma chave que foi encapsulada com o Encapsulamento de chaves AES com preenchimento PKCS #5 em uma instância do AWS CloudHSM, remova o preenchimento adicional antes de tentar usar a chave. Você pode fazer isso removendo os bytes extras em um editor de arquivo ou copiando apenas os bytes da chave em um novo buffer em seu código.

- Status da resolução: com a versão 3.1.0 de cliente e software, o AWS CloudHSM fornece opções compatíveis com os padrões para o encapsulamento de chaves AES. Para obter mais informações, consulte [Empacotamento de chaves AES](#).

Problema: o daemon do cliente requer pelo menos um endereço IP válido em seu arquivo de configuração para se conectar com sucesso ao cluster

- Impacto: se você excluir todos os HSM do seu cluster e depois adicionar outro HSM, que obterá um novo endereço IP, o daemon do cliente continuará procurando por você HSMs nos endereços IP originais.
- Solução alternativa: se você executar uma carga de trabalho intermitente, recomendamos usar o `IpAddress` argumento na [CreateHsm](#) função para definir a interface de rede elástica (ENI) com seu valor original. Observe que uma ENI é específica de uma zona de disponibilidade (AZ). A alternativa é excluir o arquivo `/opt/cloudhsm/daemon/1/cluster.info` e, em seguida, redefinir a configuração do cliente para o endereço IP do seu novo HSM. Você pode usar o comando `client -a <IP address>`. Para obter mais informações, consulte [Instalar e configurar o AWS CloudHSM cliente \(Linux\)](#) ou [Instalar e configurar o AWS CloudHSM cliente \(Windows\)](#).

Problema: havia um limite máximo de 16 KB em dados que podem ser criptografados e assinados AWS CloudHSM usando o SDK do cliente 3

- Status da resolução: os dados com extensão inferior a 16 KB continuam sendo enviados ao HSM para obter hash. Adicionamos a capacidade de aplicar hash localmente no software em dados com extensão entre 16 KB e 64 KB. O Client SDK 5 falharão explicitamente se o buffer de dados for superior a 64 KB. Você deve atualizar seu cliente e SDK(s) para uma versão superior à 5.0.0 para se beneficiar da correção.

Problema: as chaves importadas não podiam ser especificadas como não exportáveis

- Status da resolução: esse problema está corrigido. Nenhuma ação é necessária de sua parte para se beneficiar da correção.

Problema: o mecanismo padrão para o WrapKey e os unWrapKey comandos no key_mgmt_util foi removido

- **Resolução:** ao usar o WrapKey ou unWrapKey os comandos, você deve usar a -m opção para especificar o mecanismo. Consulte os exemplos no [WrapKey](#) ou nos [unWrapKey](#) artigos para obter mais informações.

Problema: se você tiver um único HSM em seu cluster, o failover do HSM não funcionará corretamente

- **Impacto:** se a única instância do HSM em seu cluster perder a conectividade, o cliente não se reconectará a ela mesmo se a instância do HSM for restaurada posteriormente.
- **Solução alternativa:** recomendamos pelo menos duas instâncias do HSM em qualquer cluster de produção. Se você usar essa configuração, você não será afetado por esse problema. Para clusters de HSM único, retorne o daemon do cliente para restaurar a conectividade.
- **Status da resolução:** este problema foi resolvido na versão 1.1.2 do cliente do AWS CloudHSM . Você deve atualizar para esse cliente para se beneficiar da correção.

Problema: se você exceder a capacidade principal do HSMs em seu cluster em um curto período de tempo, o cliente entrará em um estado de erro não tratado

- **Impacto:** quando o cliente encontra o estado de erro não processado, ele congela e deve ser reiniciado.
- **Solução alternativa:** teste seu throughput para garantir que você não esteja criando chaves de sessão a uma taxa que o cliente não consiga processar. Você pode diminuir sua taxa adicionando um HSM ao cluster ou reduzindo a criação da chave de sessão.
- **Status da resolução:** este problema foi resolvido na versão 1.1.2 do cliente do AWS CloudHSM . Você deve atualizar para esse cliente para se beneficiar da correção.

Problema: operações de resumo com chaves HMAC de tamanho maior que 800 bytes não são compatíveis.

- **Impacto:** Chaves HMAC maiores que 800 bytes podem ser geradas ou importadas para o HSM. No entanto, se você usar essa chave maior em uma operação de resumo por meio do JCE ou

key_mgmt_util, a operação falhará. Observe que, se você estiver usando PKCS11, as chaves HMAC são limitadas a um tamanho de 64 bytes.

- Solução alternativa: se você estiver usando chaves HMAC para operações de resumo no HSM, assegure-se de que o tamanho seja menor que 800 bytes.
- Status da resolução: nenhum no momento.

Problema: a ferramenta `client_info`, distribuída com o Client SDK 3, exclui o conteúdo do caminho especificado pelo argumento de saída opcional

- Impacto: todos os arquivos e subdiretórios existentes no caminho de saída especificado podem ser perdidos permanentemente.
- Solução alternativa: não use o argumento opcional `-output path` ao usar a ferramenta `client_info`.
- Status da resolução: este problema foi resolvido na [versão Client SDK 3.3.2](#). Você deve atualizar para esse cliente para se beneficiar da correção.

Problema: você recebe um erro ao executar a ferramenta de configuração do SDK 5 usando o argumento `--cluster-id` em ambientes em contêineres

Você recebe o seguinte erro ao usar o argumento `--cluster-id` com a Ferramenta de configuração:

No credentials in the property bag

Esse erro é causado por uma atualização do Instance Metadata Service versão 2 (IMDSv2). Para obter mais informações, consulte a documentação do [IMDSv2](#).

- Impacto: esse problema afetará os usuários que executam a ferramenta de configuração nas versões 5.5.0 e posteriores do SDK em ambientes em contêineres e utilizam metadados da instância EC2 para fornecer credenciais.
- Solução alternativa: defina o limite de salto de resposta do PUT para pelo menos dois. Para saber como fazer isso, consulte [Configurar as opções de metadados da instância](#).

Problema: você recebe o erro “Falha ao criar a cert/key partir do arquivo pfx fornecido.
Erro: NotPkcs 8”

- Solução alternativa: você pode converter a chave privada SSL personalizada em PKCS8 formato com o comando `openssl pkcs8 -topk8 -inform PEM -outform PEM -in ssl_private_key -out ssl_private_key_pkcs8`
- Status da resolução: esse problema foi resolvido na [versão 5.12.0 do Client SDK](#). Você deve atualizar para essa versão do cliente ou superior para se beneficiar da correção.

Problema: falha na assinatura do ECDSA com o erro “mecanismo inválido” a partir do SDK 5.16

- Impacto: as operações de assinatura do ECDSA falham ao usar funções de hash que são mais fracas do que a força da chave. Essa falha ocorre porque o [FIPS 186-5](#) exige que a função hash seja pelo menos tão forte quanto a força da chave.

Um erro semelhante a esse poderá ser exibido nos logs do cliente:

```
[cloudhsm_provider::hsm1::session::ecdsa::sign::common][[]] Digest security strength (80) is weaker than the key security strength (128)
```

- Solução alternativa: se não conseguir atualizar suas funções de hash, você poderá migrar para clusters não FIPS, que não impõem o requisito de força de hash. No entanto, recomendamos atualizar suas funções de hash para manter a conformidade com o FIPS.

Como solução alternativa, adicionamos uma opção de configuração para ignorar esse requisito. Observe que essa opção não é recomendada, pois o uso do ECDSA com funções de hash mais fracas não segue as melhores práticas de segurança. Para usar essa opção, execute o seguinte comando (substituindo `configure-cli` pela ferramenta de configuração do SDK que está sendo usada: [the section called “Sintaxe”](#)):

```
sudo /opt/cloudhsm/bin/configure-cli --enable-ecdsa-with-weak-hash-function
```

- Resolução: use uma função hash que seja pelo menos tão forte quanto sua chave ECDSA. Para obter informações sobre a função hash e os pontos fortes da chave ECDSA, consulte as Tabelas 2 e 3 no [NIST SP 800-57 Parte 1 Rev 5](#).

Problema: as operações de assinatura com dados pré-hash não apagam adequadamente os tokens de sessão no modo interativo

- Impacto: ao usar a CloudHSM CLI no modo interativo com o SDK versão 5.16.1, as operações de assinatura com dados pré-hash falham ao limpar os tokens da sessão corretamente.
- Solução alternativa: use o modo de comando único em vez do modo interativo ao realizar operações de assinatura com dados pré-hash. Isso garante a limpeza adequada do token após cada operação.
- Status da resolução: esse problema foi resolvido no CloudHSM SDK 5.16.2. Atualize para a versão 5.16.2 ou posterior para se beneficiar da correção.

Problema: o certificado de cliente padrão da biblioteca de cliente do CloudHSM expira em 31 de janeiro de 2026

- Impacto: não haverá impacto para os clientes após 31 de janeiro de 2026. [Todas as comunicações entre o cliente e o HSM são protegidas pelo TLS cliente-HSM, conforme descrito aqui.](#) O Client-HSM TLS usa owned/managed certificados de cliente que são configurados pelo cliente durante a inicialização do cluster.
- Status da resolução: Resolvido.

Problemas conhecidos para instâncias AWS CloudHSM hsm1.medium

Os problemas a seguir afetam todas as instâncias AWS CloudHSM hsm1.medium.

Tópicos

- [Problema: o HSM não consegue criar mais de 250 usuários](#)

Problema: o HSM não consegue criar mais de 250 usuários

- Solução alternativa: esse problema foi resolvido nos tipos de instância AWS CloudHSM hsm2m.medium.
- Status da resolução: nenhum no momento.

Problemas conhecidos para instâncias AWS CloudHSM hsm2m.medium

Os problemas a seguir afetam todas as instâncias AWS CloudHSM hsm2m.medium.

Tópicos

- [Problema: aumento da latência de login no hsm2m.medium](#)
- [Problema: aumento da latência de descoberta de chave no hsm2m.medium](#)
- [Problema: um CO que tente definir o atributo confiável de uma chave falhará com o Client SDK 5.12.0 e versões anteriores](#)
- [Problema: a verificação do ECDSA falhará com o Client SDK 5.12.0 e versões anteriores para clusters no modo FIPS](#)
- [Problema: somente os certificados com o formato PEM podem ser registrados como âncoras de confiança mtls com a CLI do CloudHSM](#)
- [Problema: as aplicações do cliente deixarão de processar todas as solicitações ao usar o mTLS com uma chave privada de cliente protegida por frase secreta.](#)
- [Problema: falha na replicação do usuário ao usar a CloudHSM CLI](#)
- [Problema: as operações podem falhar durante a criação do backup](#)
- [Problema: o SDK 5.8 e superior do cliente não executam novas tentativas automáticas para operações limitadas do HSM em alguns cenários em hsm2m.medium](#)
- [Problema: as operações de AES/CBC descompactamento com todas as falhas de IV zero em hsm2m.medium](#)
- [Problema: falha ao inicializar a conexão HSM durante a inicialização a frio do aplicativo em hsm2m.medium](#)

Problema: aumento da latência de login no hsm2m.medium

- Impacto: o login em hsm2m.medium segue uma interpretação excessivamente rígida dos requisitos de conformidade, o que resulta em maior latência.
- Resolução: se você criou uma nova instância hsm2m.medium ou migrou de hsm1.medium para hsm2m.medium antes de 20 de dezembro de 2025, precisará redefinir sua senha para aproveitar as melhorias de desempenho que implementamos para operações de login. Consulte a [senha de alteração](#) para obter instruções.

Problema: aumento da latência de descoberta de chave no hsm2m.medium

- **Impacto:** a instância HSM hsm2m.medium aprimorou a arquitetura de compartilhamento justo, o que resulta em um desempenho previsível mais consistente em comparação com hsm1.medium. Com o hsm1.medium, os clientes podem observar um melhor desempenho de descoberta de chave devido ao uso irregular dos recursos do HSM. No entanto, o desempenho da descoberta de chave do hsm1.medium diminuirá quando a instância do HSM for corrigida ou atualizada com um novo firmware. Esse problema afeta operações como `KeyStore.getKey()` no JCE.
- **Resolução:** Esse problema foi resolvido. Como prática recomendada, armazene em cache os resultados das operações de descoberta de chave. O armazenamento em cache reduzirá o número total de operações de descoberta de chave, pois é uma operação que consome muitos recursos no HSM. Além disso, implemente novas tentativas do lado do cliente com instabilidade e recuo exponenciais para reduzir as falhas de controle de utilização do HSM.

Problema: um CO que tente definir o atributo confiável de uma chave falhará com o Client SDK 5.12.0 e versões anteriores

- **Impacto:** qualquer usuário de CO que tente definir o atributo confiável de uma chave receberá um erro indicando `User type should be CO or CU`.
- **Resolução:** versões futuras do Client SDK resolverão esse problema. As atualizações serão anunciadas no [Histórico do documento](#) do nosso guia do usuário.

Problema: a verificação do ECDSA falhará com o Client SDK 5.12.0 e versões anteriores para clusters no modo FIPS

- **Impacto:** a operação de verificação do ECDSA realizada HSMs no modo FIPS falhará.
- **Status da resolução:** esse problema foi resolvido na [Versão 5.13.0 do Client SDK](#). Você deve atualizar para essa versão do cliente ou superior para se beneficiar da correção.

Problema: somente os certificados com o formato PEM podem ser registrados como âncoras de confiança mtls com a CLI do CloudHSM

- Impacto: certificados no formato DER não podem ser registrados como âncoras confiáveis do mTLS com a CLI do CloudHSM.
- Solução alternativa: você pode converter um certificado no formato DER para o formato PEM com o comando `openssl x509 -inform DER -outform PEM -in certificate.der -out certificate.pem`

Problema: as aplicações do cliente deixarão de processar todas as solicitações ao usar o mTLS com uma [chave privada de cliente](#) protegida por frase secreta.

- Impacto: todas as operações realizadas pela aplicação serão interrompidas, e o usuário será solicitado a inserir a frase secreta na entrada padrão várias vezes durante a vida útil da aplicação. As operações atingirão o tempo limite e falharão se a senha não for fornecida antes da duração do tempo limite da operação.
- Solução alternativa: as chaves privadas criptografadas com frase secreta não são compatíveis com o mTLS. Remova a criptografia de senha da chave privada do cliente

Problema: falha na replicação do usuário ao usar a CloudHSM CLI

- Impacto: a replicação do usuário falha nas instâncias hsm2m.medium ao usar a CloudHSM CLI. O comando `user replicate` funciona conforme o esperado nas instâncias hsm1.medium.
- Resolução: Esse problema foi resolvido.

Problema: as operações podem falhar durante a criação do backup

- Impacto: as operações como a geração de números aleatórios podem falhar em instâncias hsm2m.medium enquanto o AWS CloudHSM cria um backup.
- Resolução: para minimizar as interrupções do serviço, implemente estas melhores práticas:
 - Criar um cluster multi-HSM
 - Configurar suas aplicações para repetir as operações de cluster

Para obter mais informações sobre as melhores práticas, consulte [Melhores práticas para AWS CloudHSM](#).

Problema: o SDK 5.8 e superior do cliente não executam novas tentativas automáticas para operações limitadas do HSM em alguns cenários em hsm2m.medium

- **Impacto:** o SDK 5.8 e superior do cliente não tentará novamente algumas operações de controle de utilização do HSM
- **Solução alternativa:** siga as práticas recomendadas para arquitetar seu cluster para processar a carga e implementar novas tentativas em nível de aplicação. Estamos trabalhando em uma solução. As atualizações serão anunciadas no [Histórico do documento](#) do nosso guia do usuário.
- **Status da resolução:** esse problema foi resolvido no SDK do AWS CloudHSM cliente 5.16.2. Você deve atualizar para essa versão do cliente ou superior para se beneficiar da correção.

Problema: as operações de AES/CBC descompactamento com todas as falhas de IV zero em hsm2m.medium

- **Impacto:** ao usar o AES/CBC mecanismo para descompactar chaves usando o provedor AWS CloudHSM JCE, as operações com um IV de 16 bytes preenchido com zero falham nas instâncias hsm2m.medium, devido a uma verificação de validação adicional que não estava nas instâncias hsm1.medium.
- **Status da resolução:** estamos trabalhando em uma correção que permitirá que zero bytes IVs sejam aceitos durante AES/CBC as operações de descompactamento.

Problema: falha ao inicializar a conexão HSM durante a inicialização a frio do aplicativo em hsm2m.medium

- **Impacto:** esse problema afeta inicializações a frio, como implantações ou reinicializações de aplicativos clientes. A instância HSM hsm2m.medium aprimorou a arquitetura de compartilhamento justo, o que garante desempenho, taxa de transferência e latência mais consistentes para todos os clientes. Atualmente, no hsm1.medium, você pode observar um desempenho superior ao pretendido para a inicialização simultânea da conexão HSM. No entanto, o desempenho de inicialização da conexão hsm1.medium variará com base nas atualizações subjacentes do sistema.
- **Resolução:** siga as [melhores práticas e escalone as](#) implantações e reinicializações de aplicativos clientes para limitar a quantidade de aplicativos clientes que inicializam conexões HSM simultaneamente. Também recomendamos que você implemente novas tentativas no nível do aplicativo para a inicialização do aplicativo cliente. Além disso, inicie usando a ferramenta

configure with `--cluster-id <cluster ID>` para adicionar todos os IPs do HSM ao arquivo de configuração do cliente.

Problemas conhecidos da biblioteca PKCS #11 para AWS CloudHSM

Os problemas a seguir afetam a biblioteca PKCS #11 para AWS CloudHSM.

Tópicos

- [Problema: o encapsulamento de chave AES na versão 3.0.0 da biblioteca PKCS #11 não é IVs validado antes do uso](#)
- [Problema: o PKCS #11 SDK 2.0.4 e versões anteriores sempre usaram o padrão IV de 0xA6A6A6A6A6A6A6A6 para o encapsulamento e o desencapsulamento de chaves do AES.](#)
- [Problema: o atributo CKA_DERIVE não era compatível e, portanto, não era processado](#)
- [Problema: o atributo CKA_SENSITIVE não era compatível e, portanto, não era processado](#)
- [Problema: o hash e a assinatura em várias partes não são compatíveis.](#)
- [Problema: C_GenerateKeyPair não lida com CKA_MODULUS_BITS ou CKA_PUBLIC_EXPONENT no modelo privado de uma forma a estar em conformidade com os padrões](#)
- [Problema: os buffers das operações de API C_Encrypt e C_Decrypt não podem exceder 16 KB ao usar o mecanismo CKM_AES_GCM.](#)
- [Problema: a derivação da chave Diffie-Hellman de curva elíptica \(ECDH\) é executada parcialmente dentro do HSM.](#)
- [Problema: a verificação das assinaturas secp256k1 falha em EL6 plataformas como Cent e RHEL 6 OS6](#)
- [Problema: a sequência incorreta de chamadas de função fornece resultados indefinidos em vez de falhar](#)
- [Problema: a sessão somente leitura não é compatível com o SDK 5](#)
- [Problema: o arquivo de cabeçalho cryptoki.h é somente para Windows](#)

Problema: o encapsulamento de chave AES na versão 3.0.0 da biblioteca PKCS #11 não é IVs validado antes do uso

Se você especificar um IV com menos de 8 bytes de comprimento, ele será preenchido por bytes imprevisíveis antes de usar.

Note

Isso impacta `C_WrapKey` somente com o mecanismo `CKM_AES_KEY_WRAP`.

- Impacto: se fornecer um IV com menos de 8 bytes na versão 3.0.0 da biblioteca do PKCS #11, você poderá não conseguir desencapsular a chave.
- Soluções alternativas:
 - é altamente recomendável que você atualize para a versão 3.0.1 ou posterior da biblioteca do PKCS #11, o que aplica corretamente o comprimento do IV durante o encapsulamento de chaves do AES. Modifique o código de encapsulamento para transmitir um IV NULO ou especifique o padrão IV de `0xA6A6A6A6A6A6A6A6`. Para obter mais informações, consulte [Personalizado IVs com comprimento não compatível para AES Key Wrap](#).
 - Se você encapsulou qualquer chave com a versão 3.0.0 da biblioteca do PKCS #11 usando um IV menor que 8 bytes, entre em contato conosco para obter [suporte](#).
- Status da resolução: esse problema foi resolvido na versão 3.0.1 da biblioteca do PKCS #11. Para encapsular chaves usando o encapsulamento de chaves do AES, especifique um IV que seja NULO ou de 8 bytes de comprimento.

Problema: o PKCS #11 SDK 2.0.4 e versões anteriores sempre usaram o padrão IV de **0xA6A6A6A6A6A6A6A6** para o encapsulamento e o desencapsulamento de chaves do AES.

IVs Os fornecidos pelo usuário foram ignorados silenciosamente.

Note

Isso impacta `C_WrapKey` somente com o mecanismo `CKM_AES_KEY_WRAP`.

- Impacto:
 - se você usou o PKCS#11 SDK 2.0.4 ou uma versão anterior e um IV fornecido pelo usuário, as chaves serão encapsuladas com o IV padrão de `0xA6A6A6A6A6A6A6A6`.
 - Se você usou o PKCS#11 SDK 3.0.0 ou posterior e um IV fornecido pelo usuário, as chaves serão encapsuladas com o IV fornecido pelo usuário.

- Soluções alternativas:
 - para desencapsular chaves encapsuladas com o PKCS#11 SDK 2.0.4 ou anterior, use o padrão IV de 0xA6A6A6A6A6A6A6A6.
 - Para desencapsular chaves encapsuladas com o PKCS#11 SDK 3.0.0 ou posterior, use o IV fornecido pelo usuário.
- Status da resolução: é altamente recomendável que você modifique o código de encapsulamento e desencapsulamento para transmitir um IV NULO, ou especifique o padrão IV de 0xA6A6A6A6A6A6A6A6.

Problema: o atributo **CKA_DERIVE** não era compatível e, portanto, não era processado

- Status da resolução: implementamos correções para aceitar CKA_DERIVE caso ele seja definido como FALSE. O atributo CKA_DERIVE definido como TRUE não será comportado enquanto não começarmos a acrescentar a função de derivação de chaves no AWS CloudHSM. Você deve atualizar seu cliente e SDK(s) para a versão 1.1.1 ou superior para se beneficiar da correção.

Problema: o atributo **CKA_SENSITIVE** não era compatível e, portanto, não era processado

- Status da resolução: implementamos correções para aceitar e tratar adequadamente o atributo CKA_SENSITIVE. Você deve atualizar seu cliente e SDK(s) para a versão 1.1.1 ou superior para se beneficiar da correção.

Problema: o hash e a assinatura em várias partes não são compatíveis.

- Impacto: C_DigestUpdate e C_DigestFinal não estão implementados. C_SignFinal também não é implementado e apresentará falha com CKR_ARGUMENTS_BAD para um buffer não NULL.
- Solução alternativa: faça o hash dos dados em seu aplicativo e use-os AWS CloudHSM somente para assinar o hash.
- Status da resolução: estamos corrigindo o cliente e implementando corretamente SDKs o hashing de várias partes. As atualizações serão anunciadas no fórum do AWS CloudHSM e na página de histórico de versões.

Problema: C_GenerateKeyPair não lida com **CKA_MODULUS_BITS** ou **CKA_PUBLIC_EXPONENT** no modelo privado de uma forma a estar em conformidade com os padrões

- Impacto: C_GenerateKeyPair deve retornar CKA_TEMPLATE_INCONSISTENT quando o modelo privado contém CKA_MODULUS_BITS ou CKA_PUBLIC_EXPONENT. Em vez disso, ele gera uma chave privada para a qual todos os campos de uso são definidos como FALSE. A chave não pode ser usada.
- Solução alternativa: recomendamos que o seu aplicativo verifique os valores dos campos de uso, além do código de erros.
- Status da resolução: estamos implementando correções para retornar a mensagem de erro adequada quando um modelo de chave privada incorreto é usado. A biblioteca PKCS#11 atualizada será anunciada na página de histórico de versões.

Problema: os buffers das operações de API C_Encrypt e C_Decrypt não podem exceder 16 KB ao usar o mecanismo **CKM_AES_GCM**.

AWS CloudHSM não suporta criptografia AES-GCM de várias partes.

- Impacto: não é possível usar o mecanismo CKM_AES_GCM para criptografar dados maiores que 16 KB.
- Solução alternativa: use um mecanismo alternativo como o CKM_AES_CBC , CKM_AES_CBC_PAD ou divida os dados em partes e criptografe cada parte usando AES_GCM individualmente. Se você estiver usando AES_GCM, deverá gerenciar a divisão de seus dados e a criptografia subsequente. AWS CloudHSM não executa criptografia AES-GCM em várias partes para você. Observe que o FIPS requer que o vetor de inicialização (IV) para AES-GCM seja gerado no HSM. Portanto, o IV para cada parte dos dados criptografados em AES-GCM será diferente.
- Status da resolução: vamos corrigir o SDK para falhar explicitamente se o buffer de dados for muito grande. Retornaremos CKR_MECHANISM_INVALID para as operações de API C_EncryptUpdate e C_DecryptUpdate. Estamos avaliando alternativas para oferecer suporte a buffers maiores sem depender da criptografia em várias partes. As atualizações serão anunciadas no AWS CloudHSM fórum e na página do histórico de versões.

Problema: a derivação da chave Diffie-Hellman de curva elíptica (ECDH) é executada parcialmente dentro do HSM.

Sua chave privada EC permanece no HSM em todos os momentos, mas o processo de derivação de chaves é realizado em várias etapas. Consequentemente, os resultados intermediários de cada etapa estão disponíveis no cliente.

- **Impacto:** no Client SDK 3, a chave derivada por meio do mecanismo CKM_ECDH1_DERIVE fica disponível primeiro no cliente e, em seguida, é importada para o HSM. Um identificador de chave é retornado à aplicação.
- **Solução alternativa:** se você estiver implementando o SSL/TLS Offload in AWS CloudHSM, essa limitação pode não ser um problema. Se a aplicação precisar da sua chave para permanecer continuamente em um limite FIPS, é recomendável o uso de um protocolo alternativo que não dependa da derivação de chaves ECDH.
- **Status da resolução:** agora o SDK 5.16 é compatível com ECDH com derivação de chave, que é executada inteiramente dentro do HSM.

Problema: a verificação das assinaturas secp256k1 falha em EL6 plataformas como Cent e RHEL 6 OS6

Isso ocorre porque a biblioteca PKCS #11 do CloudHSM evita uma chamada de rede durante a inicialização da operação de verificação usando o OpenSSL para verificar dados da curva do EC. Como o SecP256k1 não é suportado pelo pacote OpenSSL padrão nas plataformas, a inicialização falha. EL6

- **Impacto:** a verificação da assinatura SECP256k1 falhará nas plataformas. EL6 Haverá falha na chamada para verificar com um erro CKR_HOST_MEMORY.
- **Solução alternativa:** recomendamos usar o Amazon Linux 1 ou qualquer EL7 plataforma se seu aplicativo PKCS #11 precisar verificar as assinaturas secp256k1. Como alternativa, atualize para uma versão do pacote OpenSSL que ofereça suporte à curva secp256k1.
- **Status da resolução:** estamos implementando correções para voltar para o HSM se a validação da curva local não estiver disponível. A biblioteca PKCS#11 atualizada será anunciada na página de [histórico de versões](#).

Problema: a sequência incorreta de chamadas de função fornece resultados indefinidos em vez de falhar

- Impacto: se você chamar uma sequência incorreta de funções, o resultado final será incorreto, mesmo que as chamadas de função individuais retornem com sucesso. Por exemplo, os dados descritos podem não corresponder ao texto simples original ou as assinaturas podem falhar na verificação. Esse problema afeta as operações de parte única e de várias partes.

Exemplos de sequências de funções incorretas:

- C_EncryptInit/C_EncryptUpdatesseguido por C_Encrypt
- C_DecryptInit/C_DecryptUpdatesseguido por C_Decrypt
- C_SignInit/C_SignUpdatesseguido por C_Sign
- C_VerifyInit/C_VerifyUpdatesseguido por C_Verify
- C_FindObjectsInitseguido por C_FindObjectsInit
- Solução alternativa: seu aplicativo deve, em conformidade com a especificação PKCS #11, usar a sequência correta de chamadas de função para operações de uma e várias partes. Seu aplicativo não deve depender da biblioteca PKCS #11 do CloudHSM para retornar um erro nessa circunstância.

Problema: a sessão somente leitura não é compatível com o SDK 5

- Problema: o SDK 5 não oferece suporte à abertura de sessões somente leitura com o C_OpenSession.
- Impacto: se você tentar chamar C_OpenSession sem fornecer CKF_RW_SESSION, a chamada falhará com o erro CKR_FUNCTION_FAILED.
- Solução alternativa: ao abrir uma sessão, você deve passar os CKF_SERIAL_SESSION | CKF_RW_SESSION sinalizadores para a chamada da função C_OpenSession.

Problema: o arquivo de cabeçalho **cryptoki.h** é somente para Windows

- Problema: com as versões 5.0.0 a 5.4.0 do SDK do AWS CloudHSM Cliente 5 no Linux, o arquivo de cabeçalho só /opt/cloudhsm/include/pkcs11/cryptoki.h é compatível com sistemas operacionais Windows.
- Impacto: você pode encontrar problemas ao tentar incluir esse arquivo de cabeçalho em seu aplicativo em sistemas operacionais baseados em Linux.

- Status da resolução: atualize para o AWS CloudHSM Client SDK 5 versão 5.4.1 ou superior, que inclui uma versão compatível com Linux desse arquivo de cabeçalho.

Problemas conhecidos do JCE SDK para AWS CloudHSM

Os problemas a seguir afetam o SDK do JCE para AWS CloudHSM

Tópicos

- [Problema: ao trabalhar com pares de chaves assimétricas, você vê a capacidade de chaves ocupada mesmo quando não está explicitamente criando ou importando chaves](#)
- [Problema: O JCE KeyStore é somente para leitura](#)
- [Problema: os buffers para criptografia AES-GCM não podem exceder 16.000 bytes.](#)
- [Problema: a derivação da chave Diffie-Hellman de curva elíptica \(ECDH\) é executada parcialmente dentro do HSM.](#)
- [Problema: KeyGenerator e interpreta KeyAttribute incorretamente o parâmetro de tamanho da chave como número de bytes em vez de bits](#)
- [Problema: o Client SDK 5 emite o aviso “Ocorreu uma operação ilegal de acesso reflexivo”](#)
- [Problema: o pool de sessões do JCE está esgotado](#)
- [Problema: vazamento de memória do Client SDK 5 com operações getKey](#)

Problema: ao trabalhar com pares de chaves assimétricas, você vê a capacidade de chaves ocupada mesmo quando não está explicitamente criando ou importando chaves

- Impacto: esse problema pode fazer HSMs com que você fique sem espaço de chave inesperadamente e ocorre quando seu aplicativo usa um objeto de chave JCE padrão para operações de criptografia em vez de um objeto. `CaviumKey` Quando você usa um objeto de chave JCE padrão, o `CaviumProvider` importa implicitamente essa chave para o HSM como uma chave de sessão e não exclui essa chave até que o aplicativo seja encerrado. Como resultado, as chaves se acumulam enquanto o aplicativo está em execução e podem fazer HSMs com que você fique sem espaço livre na chave, congelando seu aplicativo.
- Solução alternativa: ao usar a classe `CaviumSignature`, a classe `CaviumCipher`, a classe `CaviumMac` ou a classe `CaviumKeyAgreement`, você deverá fornecer a chave um `CaviumKey` em vez de um objeto de chave JCE padrão.

É possível converter manualmente uma chave normal em um CaviumKey usando a classe [ImportKey](#) e depois excluir manualmente a chave após a conclusão da operação.

- Status da resolução: estamos atualizando o CaviumProvider para gerenciar corretamente as importações implícitas. A correção será anunciada na página de histórico de versões quando estiver disponível.

Problema: O JCE KeyStore é somente para leitura

- Impacto: não é possível armazenar um tipo de objeto que não seja compatível com o HSM no repositório de chaves do JCE. Sobretudo, você não pode armazenar certificados no repositório de chaves. Isso impede a interoperabilidade com ferramentas como jarsigner, que espera encontrar o certificado no repositório de chaves.
- Solução alternativa: você pode reprocessar seu código para carregar certificados de arquivos locais ou de um local do bucket do S3 em vez de carregá-los do repositório de chaves.
- Status da resolução: você pode usar o AWS CloudHSM keystore para armazenar certificados.

Problema: os buffers para criptografia AES-GCM não podem exceder 16.000 bytes.

Além disso, não há suporte para a criptografia AES-GCM em várias partes.

- Impacto: não é possível usar o AES-GCM para criptografar dados maiores que 16.000 bytes.
- Solução alternativa: use um mecanismo alternativo como o AES-CBC ou divida os dados em partes e criptografe cada parte individualmente. Se você dividir os dados, gerencie o texto cifrado dividido e sua descryptografia. Como o FIPS exige que o vetor de inicialização (IV) do AES-GCM seja gerado no HSM, o IV para cada AES-GCM-encrypted dado será diferente.
- Status da resolução: vamos corrigir o SDK para falhar explicitamente se o buffer de dados for muito grande. Estamos avaliando alternativas para oferecer suporte a buffers maiores sem depender da criptografia em várias partes. As atualizações serão anunciadas no fórum do AWS CloudHSM e na página de histórico de versões.

Problema: a derivação da chave Diffie-Hellman de curva elíptica (ECDH) é executada parcialmente dentro do HSM.

Sua chave privada EC permanece no HSM em todos os momentos, mas o processo de derivação de chaves é realizado em várias etapas. Consequentemente, os resultados intermediários de cada

etapa estão disponíveis no cliente. Um exemplo de derivação de chave ECDH está disponível nos [exemplos de código Java](#).

- Impacto: o Client SDK 3 adiciona a funcionalidade ECDH ao JCE. Quando você usa a `KeyAgreement` classe para derivar uma `SecretKey`, ela fica disponível primeiro no cliente e depois é importada para o HSM. Um identificador de chave é retornado à aplicação.
- Solução alternativa: se você estiver implementando o SSL/TLS Offload in AWS CloudHSM, essa limitação pode não ser um problema. Se a aplicação precisar da sua chave para permanecer continuamente em um limite FIPS, é recomendável o uso de um protocolo alternativo que não dependa da derivação de chaves ECDH.
- Status da resolução: agora o SDK 5.16 é compatível com ECDH com derivação de chave, que é executada inteiramente dentro do HSM.

Problema: `KeyGenerator` interpreta `KeyAttribute` incorretamente o parâmetro de tamanho da chave como número de bytes em vez de bits

Ao gerar uma chave usando a `init` função da [KeyGenerator classe](#) ou o `SIZE` atributo do [AWS CloudHSM KeyAttribute enum](#), a API espera incorretamente que o argumento seja o número de bytes da chave, quando deveria ser o número de bits da chave.

- Impacto: as versões 5.4.0 a 5.4.2 do SDK do cliente esperam incorretamente que o tamanho da chave seja fornecido ao especificado em bytes. APIs
- Solução alternativa: converta o tamanho da chave de bits em bytes antes de usar a `KeyGenerator` classe ou `KeyAttribute` enum para gerar chaves usando o provedor AWS CloudHSM JCE se estiver usando as versões 5.4.0 a 5.4.2 do SDK do Cliente.
- Status da resolução: atualize a versão do SDK do cliente para 5.5.0 ou posterior, o que inclui uma correção para esperar corretamente os tamanhos das chaves em bits ao usar a `KeyGenerator` classe ou o `KeyAttribute` enum para gerar chaves.

Problema: o Client SDK 5 emite o aviso “Ocorreu uma operação ilegal de acesso reflexivo”

Ao usar o Client SDK 5 com Java 11, o CloudHSM emite o seguinte aviso de Java:

```
WARNING: An illegal reflective access operation has occurred
```

```
WARNING: Illegal reflective access by
  com.amazonaws.cloudhsm.jce.provider.CloudHsmKeyStore (file:/opt/cloudhsm/java/
cloudhsm-jce-5.6.0.jar) to field java.security .KeyStore.keyStoreSpi
WARNING: Please consider reporting this to the maintainers of
  com.amazonaws.cloudhsm.jce.provider.CloudHsmKeyStore
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective
  access operations
WARNING: All illegal access operations will be denied in a future release
```

Esse problema foi corrigido no Client SDK versão 5.8 e posterior.

Problema: o pool de sessões do JCE está esgotado

Impacto: talvez você não consiga realizar operações no JCE depois de ver a seguinte mensagem:

```
com.amazonaws.cloudhsm.jce.jni.exception.InternalException: There are too many
  operations
happening at the same time: Reached max number of sessions in session pool: 1000
```

Soluções alternativas:

- Reinicie seu aplicativo JCE se você estiver enfrentando um impacto.
- Ao realizar uma operação, talvez seja necessário concluir a operação JCE antes de perder a referência à operação.

Note

Dependendo da operação, pode ser necessário um método de preenchimento.

Operation	Método(s) de preenchimento
Cifra	doFinal() no modo criptografar ou descriptografar
	wrap() no modo encapsular
	unwrap() no modo desencapsular

Operation	Método(s) de preenchimento
KeyAgreement	<code>generateSecret()</code> ou <code>generateSecret(String)</code>
KeyPairGenerator	<code>generateKeyPair()</code> , <code>genKeyPair()</code> ou <code>reset()</code>
KeyStore	Nenhum método é necessário
Mac	<code>doFinal()</code> ou <code>reset()</code>
MessageDigest	<code>digest()</code> ou <code>reset()</code>
SecretKeyFactory	Nenhum método é necessário
SecureRandom	Nenhum método é necessário
Signature	<code>sign()</code> no modo de sinal <code>verify()</code> no modo de verificação

Status da resolução: resolvemos esse problema no Client SDK 5.9.0 e versões posteriores. Para corrigir esse problema, atualize seu Client SDK para uma dessas versões.

Problema: vazamento de memória do Client SDK 5 com operações `getKey`

- Impacto: a operação da API `getKey` tem um vazamento de memória no JCE nas versões 5.10.0 e anteriores do Client SDK. Se você estiver usando a API `getKey` várias vezes em sua aplicação, isso aumentará o crescimento da memória e, conseqüentemente, aumentará o espaço ocupado pela memória em sua aplicação. Com o tempo, isso pode causar erros de controle de utilização ou exigir que a aplicação seja reiniciada.
- Solução alternativa: recomendamos a atualização para o Client SDK 5.11.0. Se isso não puder ser feito, recomendamos não chamar a API `getKey` várias vezes em sua aplicação. Em vez disso, reutilize a chave retornada anteriormente da operação `getKey` anterior, tanto quanto possível.
- Status da resolução: atualize a versão do Client SDK para 5.11.0 ou posterior, o que inclui uma correção para esse problema.

Problemas conhecidos do OpenSSL Dynamic Engine para AWS CloudHSM

Estes são os problemas conhecidos do OpenSSL Dynamic Engine para AWS CloudHSM.

Tópicos

- [Problema: você não pode instalar o AWS CloudHSM OpenSSL Dynamic Engine no RHEL 6 e no Cent OS6](#)
- [Problema: por padrão, somente o descarregamento de RSA para o HSM é compatível.](#)
- [Problema: não há suporte para criptografia e descryptografia RSA com preenchimento OAEP usando uma chave no HSM.](#)
- [Problema: somente a geração de chave privada de chaves RSA e ECC é descarregada para o HSM.](#)
- [Problema: você não pode instalar o OpenSSL Dynamic Engine para Client SDK 3 no RHEL 8, CentOS 8 ou Ubuntu 18.04 LTS](#)
- [Problema: descontinuação do uso do SHA-1 Sign and Verify no RHEL 9 \(9.2+\)](#)
- [Problema: o AWS CloudHSM OpenSSL Dynamic Engine é incompatível com o provedor FIPS para OpenSSL v3.x](#)
- [Problema: falha no SSL/TLS descarregamento com conjuntos de criptografia ECDSA no TLS 1.0 e no TLS 1.1 a partir do SDK 5.16](#)

Problema: você não pode instalar o AWS CloudHSM OpenSSL Dynamic Engine no RHEL 6 e no Cent OS6

- Impacto: o OpenSSL Dynamic Engine [oferece suporte apenas para OpenSSL 1.0.2 \[f+\]](#). Por padrão, o RHEL 6 e o CentOS 6 são fornecidos com o OpenSSL 1.0.1.
- Solução alternativa: atualize a biblioteca OpenSSL no RHEL 6 e no CentOS 6 para a versão 1.0.2 [f+].

Problema: por padrão, somente o descarregamento de RSA para o HSM é compatível.

- Impacto: para maximizar o desempenho, o SDK não está configurado para descarregar funções adicionais, como a geração aleatória de números ou operações de EC-DH.
- Solução alternativa: entre em contato conosco por meio de um caso de suporte se precisar descarregar operações adicionais.

- Status da resolução: estamos adicionando suporte ao SDK para configurar as opções de descarregamento por meio de um arquivo de configuração. A atualização será anunciada na página de histórico de versões, quando estiver disponível.

Problema: não há suporte para criptografia e descryptografia RSA com preenchimento OAEP usando uma chave no HSM.

- Impacto: qualquer chamada para criptografia e decodificação RSA com preenchimento OAEP falha com um erro. divide-by-zero Isso ocorre porque o mecanismo dinâmico OpenSSL chama a operação localmente usando o arquivo PEM falso em vez de descarregar a operação para o HSM.
- Solução alternativa: realize esse procedimento usando a [Biblioteca PKCS #11 para AWS CloudHSM Client SDK 5](#) ou a [Provedor de JCE para AWS CloudHSM Client SDK 5](#).
- Status da resolução: estamos adicionando suporte ao SDK para descarregar corretamente essa operação. A atualização será anunciada na página de histórico de versões, quando estiver disponível.

Problema: somente a geração de chave privada de chaves RSA e ECC é descarregada para o HSM.

Para qualquer outro tipo de chave, o mecanismo AWS CloudHSM OpenSSL não é usado para processamento de chamadas. Em vez disso, o mecanismo OpenSSL local será usado. Isso gera uma chave localmente no software.

- Impacto: como o failover é silencioso, não há indicação de que você não recebeu uma chave gerada com segurança no HSM. Você verá um rastreamento de saída que contém a string ".+++++" se a chave for gerada localmente pelo OpenSSL no software. Esse rastreamento está ausente quando a operação é descarregada para o HSM. Como a chave não é gerada ou armazenada no HSM, ela não estará disponível para uso futuro.
- Solução alternativa: use o mecanismo OpenSSL somente para tipos de chave compatíveis. Para todos os outros tipos de chave, use PKCS#11 ou JCE nas aplicações ou use key_mgmt_util na CLI.

Problema: você não pode instalar o OpenSSL Dynamic Engine para Client SDK 3 no RHEL 8, CentOS 8 ou Ubuntu 18.04 LTS

- Impacto: por padrão, o RHEL 8, o CentOS 8 e o Ubuntu 18.04 LTS vêm com uma versão do OpenSSL que não é compatível com o OpenSSL Dynamic Engine para Client SDK 3.
- Solução alternativa: use uma plataforma Linux que ofereça suporte ao OpenSSL Dynamic Engine. Para obter mais informações sobre as plataformas compatíveis, consulte [Plataformas compatíveis](#).
- Status da resolução: AWS CloudHSM suporta essas plataformas com o OpenSSL Dynamic Engine for Client SDK 5. Para obter mais informações, consulte [Plataformas compatíveis](#) e [OpenSSL Dynamic Engine](#).

Problema: descontinuação do uso do SHA-1 Sign and Verify no RHEL 9 (9.2+)

- Impacto: o uso do resumo de mensagens SHA-1 para fins criptográficos foi descontinuado no RHEL 9 (9.2+). Como resultado, as operações de assinatura e verificação com SHA-1 usando o OpenSSL Dynamic Engine falharão.
- Solução alternativa: [se seu cenário exigir o uso de SHA-1 para assinaturas criptográficas signing/verifying existentes ou de terceiros, consulte Aprimorando a segurança do RHEL: entendendo a depreciação do SHA-1 no RHEL 9 \(9.2+\) e no RHEL 9 \(9.2+\) para obter mais detalhes.](#)

Problema: o AWS CloudHSM OpenSSL Dynamic Engine é incompatível com o provedor FIPS para OpenSSL v3.x

- Impacto: você receberá uma mensagem de erro se tentar utilizar o AWS CloudHSM OpenSSL Dynamic Engine quando o provedor FIPS estiver habilitado para as versões 3.x do OpenSSL.
- Solução alternativa: para usar o AWS CloudHSM OpenSSL Dynamic Engine com as versões 3.x do OpenSSL, verifique se o provedor “padrão” está configurado. Leia mais sobre o provedor padrão no [site do OpenSSL](#).

Problema: falha no SSL/TLS descarregamento com conjuntos de criptografia ECDSA no TLS 1.0 e no TLS 1.1 a partir do SDK 5.16

- Impacto: as tentativas de conexão usando TLS 1.0 ou TLS 1.1 falham porque essas versões usam SHA-1 para assinatura, o que não atende aos requisitos do [FIPS 186-5](#).

- Solução alternativa: se não conseguir atualizar as versões do TLS imediatamente, você poderá migrar para clusters não FIPS, que não impõem o requisito de força de hash. No entanto, recomendamos a atualização para o TLS 1.2 ou TLS 1.3 para manter a conformidade com o FIPS e as melhores práticas de segurança.
- Resolução: atualize sua implementação para usar o TLS 1.2 ou TLS 1.3. A Internet Engineering Task Force (IETF) [descontinuou](#) o TLS 1.0 e o TLS 1.1 devido a questões de segurança.

Problemas conhecidos do Key Storage Provider (KSP) para AWS CloudHSM

Estes são os problemas conhecidos do Key Storage Provider (KSP) para AWS CloudHSM.

Tópicos

- [Problema: falha na verificação de um armazenamento de certificados](#)
- [Problema: inconsistência do nome do contêiner no armazenamento de certificados ao usar o modo de SDK3 compatibilidade para o Client SDK 5](#)

Problema: falha na verificação de um armazenamento de certificados

Ao usar as versões 5.14 e 5.15 do Client SDK, chamar `certutil -store my CERTIFICATE_SERIAL_NUMBER` gera o seguinte erro:

```
ERROR: Could not verify certificate public key against private key
```

- Impacto: você não pode usar `certutil` para validar um armazenamento de certificados criado com o Client SDK 5.
- Solução alternativa: valide o par de chaves associado ao certificado assinando um arquivo usando a chave privada e verificando a assinatura usando a chave pública. Isso pode ser feito usando a Microsoft SignTool seguindo as etapas fornecidas [aqui](#).
- Status da resolução: estamos trabalhando para adicionar suporte para verificar certificados usando `certutil`. A correção será anunciada na página de histórico de versões quando estiver disponível.

Problema: inconsistência do nome do contêiner no armazenamento de certificados ao usar o modo de SDK3 compatibilidade para o Client SDK 5

Ao usar o `certutil -store my CERTIFICATE_SERIAL_NUMBER` comando para visualizar certificados cujos arquivos de referência de chave foram gerados usando o comando [generate-file](#) na AWS CLI versão 5.16.0, ocorre o seguinte erro:

```
ERROR: Container name inconsistent: CONTAINER_NAME
```

Esse erro ocorre porque há uma incompatibilidade entre o nome do contêiner armazenado no certificado e o nome do arquivo de referência da chave gerado pela CloudHSM CLI.

- Impacto: apesar desse erro, os certificados e suas chaves associadas permanecem totalmente funcionais. Todas as aplicações que usam esses certificados continuarão funcionando normalmente.
- Solução alternativa: para resolver esse erro, renomeie o nome do arquivo de referência da chave para Nome de contêiner simples ou exclusivo. Consulte o seguinte exemplo de saída do comando `certutil -store my`

```
Subject: CN=www.website.com, OU=Organizational-Unit, O=Organization, L=City, S=State, C=US
Non-root Certificate
Cert Hash(sha1): 1add52
Key Container = 7e3c-b2f5
Simple container name: tq-3daacd89
Unique container name: tq-3daacd89
ERROR: Container name inconsistent: 7e3c-b2f5
```

Por padrão, os principais arquivos de referência serão armazenados em `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`

1. Renomeie o arquivo de referência da chave para o nome simples do contêiner.
 2. Repare o repositório de certificados com o novo nome do contêiner de chaves. Consulte as etapas 12 a 14 na [Migração do KSP](#) para obter mais detalhes.
- Status da resolução: este problema foi resolvido no Client SDK versão 5.16.1. Para resolver esse problema, atualize seu Client SDK para a versão 5.16.1 ou posterior.

Problemas conhecidos do OpenSSL Provider para AWS CloudHSM

Esses são os problemas conhecidos do OpenSSL Provider for. AWS CloudHSM

Tópicos

- [Problema: erros na CLI do OpenSSL quando usada com o OpenSSL Provider](#)

Problema: erros na CLI do OpenSSL quando usada com o OpenSSL Provider

- Impacto: a integração com o OpenSSL CLI não é atualmente suportada pelo OpenSSL Provider. AWS CloudHSM Consulte [the section called “Descarregue no Linux com o OpenSSL Provider”](#) as integrações compatíveis.
- Resolução: use as integrações compatíveis listadas em [the section called “Descarregue no Linux com o OpenSSL Provider”](#). Todas as atualizações no suporte ao OpenSSL CLI serão anunciadas na página do histórico de versões.

Problemas conhecidos para instâncias do Amazon EC2 executando o Amazon Linux 2 com AWS CloudHSM

Os seguintes problemas afetam AWS CloudHSM as instâncias do Amazon EC2 que estão sendo executadas no Amazon Linux 2.

Problema: o Amazon Linux 2 versão 2018.07 usa um **ncurses** pacote atualizado (versão 6) que atualmente é incompatível com o AWS CloudHSM SDKs

[Você vê o seguinte erro retornado ao executar o AWS CloudHSMcloudhsm_mgmt_util ou key_mgmt_util:](#)

```
/opt/cloudhsm/bin/cloudhsm_mgmt_util: error while loading shared libraries:  
libncurses.so.5: cannot open shared object file: No such file or directory
```

- Impacto: as instâncias executadas no Amazon Linux 2 versão 2018.07 não poderão usar todos os AWS CloudHSM utilitários.
- Solução alternativa: emita o seguinte comando em suas instâncias do Amazon Linux 2 EC2 para instalar o pacote ncurses compatível (versão 5):

```
sudo yum update && yum install ncurses-compat-libs
```

- Status da resolução: este problema foi resolvido na versão 1.1.2 do cliente do AWS CloudHSM . Você deve atualizar para esse cliente para se beneficiar da correção.

Problemas conhecidos na integração de aplicativos de terceiros com AWS CloudHSM

Os problemas a seguir afetam AWS CloudHSM a integração com aplicativos de terceiros.

Problema: o Client SDK 3 não é compatível com o Oracle ao definir o atributo **CKA_MODIFIABLE** do PKCS #11 durante a geração da chave mestra

Esse limite é definido na biblioteca PKCS #11. Para obter mais informações, consulte a anotação 1 em [Atributos do PKCS #11 compatíveis](#).

- Impacto: falha na criação da chave mestra do Oracle.
- Solução alternativa: defina a variável de ambiente especial `CLOUDHSM_IGNORE_CKA_MODIFIABLE_FALSE` como `TRUE` ao criar uma chave mestra. Essa variável de ambiente só é necessária para geração de chave mestra e não é necessário usá-la para qualquer outra coisa. Por exemplo, você usaria essa variável de ambiente para a primeira chave mestra criada e, depois, você só a usaria novamente se quisesse mudar a edição da chave mestra. Para obter mais informações, consulte [Gerar a chave de criptografia mestra do Oracle TDE](#).
- Status da resolução: estamos melhorando o firmware do HSM para oferecer suporte total ao atributo `CKA_MODIFIABLE`. As atualizações serão anunciadas no AWS CloudHSM fórum e na página do histórico de versões

Problemas conhecidos de modificação AWS CloudHSM do cluster

Os problemas a seguir afetam os clientes que tentam usar a API `modify-cluster` para alterar o tipo de HSM de um cluster.

Tópicos

- [Problema: a latência de login aumenta devido ao aumento de iterações PBKDF2](#)
- [Problema: não é possível modificar o tipo de HSM devido à criação da chave de token](#)

Problema: a latência de login aumenta devido ao aumento de iterações PBKDF2

- Impacto: os clusters com uma grande quantidade de usuários passarão por um período de migração prolongado. Isso se deve às mudanças no processo de restauração de backup que executa PBKDF2 operações por usuário ao restaurar um backup hsm1.medium em hsm2m.medium pela primeira vez.
- Status da resolução: se você criou uma nova instância hsm2m.medium ou migrou de hsm1.medium para hsm2m.medium antes de 20 de dezembro de 2025, precisará redefinir sua senha para aproveitar as melhorias de desempenho que implementamos para operações de login. Consulte a [senha de alteração](#) para obter instruções.

Problema: não é possível modificar o tipo de HSM devido à criação da chave de token

- Impacto: os clientes que executam workloads baseadas em chaves de token não conseguirão iniciar a migração. Isso é feito porque o HSM será colocado em um modo de gravação limitada para evitar cenários de perda de dados durante a modificação do tipo de HSM.
- Solução alternativa: pare de criar e excluir chaves de token e aguarde 7 dias. Como alternativa, entre em contato com o suporte se você
 - Não consegue lidar com o bloqueio de migrações de chaves de token e não pode fazer uma blue/green implantação.
 - Puder lidar com o bloqueio de operações de chave de token durante a migração, mas não puder esperar o período completo de 7 dias.
- Status da resolução: esse problema foi resolvido. Os clientes que executam workloads baseadas em chaves de token agora podem começar a migração. As criações e exclusões de chaves de token serão bloqueadas durante a migração.

Problemas conhecidos de falha de operação usando a versão 5.12.0 do AWS CloudHSM cliente no hsm2.medium

Os seguintes problemas afetam AWS CloudHSM o uso da versão AWS CloudHSM 5.12.0 do cliente

Problema: erro durante a operação get-attribute

Se estiver migrando de hsm1.medium para hsm2m.medium e usando o CloudHSM Client SDK 5.12.0, poderão ocorrer erros relacionados ao tratamento de atributos.

Você poderá ver a seguinte mensagem de erro nos logs do cliente: `Error in deserialization of data: Invalid integer conversion`

Impacto: as operações abaixo falharão usando o cliente versão 5.12.0

- No SDK PKCS #11, as chamadas para `C_` falham `GetAttributeValue`
- Na CloudHSM CLI, o comando `key list` não mostra atributos na saída
- Na CloudHSM CLI, o arquivo de geração de chave pode falhar para chaves geradas usando `hsm1.medium`

Resolução: recomendamos a atualização para a versão mais recente do SDK, que resolve esse problema.

AWS CloudHSM Falhas de sincronização de teclas do Client SDK 3

No Client SDK 3, se a sincronização do lado do cliente falhar, AWS CloudHSM faça o melhor possível para limpar todas as chaves indesejadas que possam ter sido criadas (e agora são indesejadas). Esse processo envolve a remoção imediata do material de chave indesejado ou a marcação do material indesejado para remoção posterior. Em ambos os casos, a resolução não exige nenhuma ação de sua parte. No caso raro de AWS CloudHSM não conseguir remover e não poder marcar material de chave indesejado, você deve excluir o material de chave.

Problema: você tenta uma operação de geração, importação ou desagrupamento da chave de token e vê erros que especificam uma falha na marca de exclusão.

```
2018-12-24T18:28:54Z liquidSecurity ERR: print_node_ts_status:
[create_object_min_nodes]Key: 264617 failed to tombstone on node:1
```

Causa: não AWS CloudHSM foi possível remover e marcar material de chave indesejado.

Resolução: um HSM em seu cluster contém material de chave indesejado que não está marcado como indesejado. Você deve remover o material de chave manualmente. Para excluir manualmente o material de chave indesejado, use `key_mgmt_util` (KMU) ou uma API da biblioteca PKCS #11 ou do provedor JCE. Para ter mais informações, consulte [deleteKey](#) ou [Cliente SDKs](#).

Para tornar as chaves de token mais duráveis, AWS CloudHSM falha nas operações de criação de chaves que não são bem-sucedidas no número mínimo HSMs especificado nas configurações de

sincronização do lado do cliente. Para obter mais informações, consulte [Sincronização de chaves no AWS CloudHSM](#).

AWS CloudHSM O SDK 3 do cliente verifica o desempenho do HSM com a ferramenta pkpspeed

Este tópico descreve como verificar o desempenho do módulo AWS CloudHSM de segurança de hardware (HSM) com o Client SDK 3.

Para verificar o desempenho dos HSMs em seu AWS CloudHSM cluster, você pode usar a ferramenta `pkpspeed` (Linux) ou `pkpspeed_blocking` (Windows) incluída no Client SDK 3. A ferramenta `pkpspeed` é executada em condições ideais e chama diretamente o HSM para executar operações sem passar por um SDK, como PKCS11. Recomendamos testar a carga de seu aplicativo de forma independente para determinar suas necessidades de escalabilidade. Não recomendamos a execução dos seguintes testes: Random (I), ModExp (R) e EC point mul (Y).

Para obter mais informações sobre como instalar o cliente em uma instância do EC2 para Linux, consulte [Instale e configure o AWS CloudHSM cliente para CMU \(Linux\)](#). Para obter mais informações sobre como instalar o cliente em uma instância do Windows, consulte [Instale e configure o AWS CloudHSM cliente para CMU \(Windows\)](#).

Depois de instalar e configurar o AWS CloudHSM cliente, execute o comando a seguir para iniciá-lo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```


RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe C:  
\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Se você já instalou o software cliente, talvez seja necessário fazer download e instalar a versão mais recente para obter a pkpspeed. Você pode encontrar a ferramenta pkpspeed em /opt/cloudhsm/bin/pkpspeed no Linux ou C:\Program Files\Amazon\CloudHSM\ no Windows.

Para usar pkpspeed, execute o comando pkpspeed ou pkpspeed_blocking.exe, especificando o nome do usuário e a senha de um usuário de criptografia (CU) no HSM. Em seguida, defina as opções a serem usadas, considerando as seguintes recomendações.

Teste das recomendações

- Para testar o desempenho do sinal RSA e verificar as operações, escolha a cifra RSA_CRT em Linux ou a opção B no Windows. Não escolha RSA (opção A no Windows). As cifras são equivalentes, mas RSA_CRT são otimizadas para o desempenho.
- Comece com um pequeno número de tópicos. Para testar o desempenho do AES, um tópico normalmente é suficiente para mostrar o máximo desempenho. Para testar o desempenho do RSA (RSA_CRT), normalmente são suficientes três ou quatro tópicos.

Opções configuráveis para a ferramenta pkpspeed

- Modo FIPS: AWS CloudHSM está sempre no modo FIPS (consulte [AWS CloudHSM FAQs](#) para obter detalhes). Isso pode ser verificado usando as ferramentas da CLI, conforme documentado no Guia do AWS CloudHSM usuário, e executando o [Obtenha informações de hardware para cada HSM em um AWS CloudHSM cluster com CMU](#) comando que indicará o status do modo FIPS.
- Tipo de teste (com bloqueio x sem bloqueio): especifica como as operações são executadas de forma segmentada. Você provavelmente obterá números melhores usando o sistema sem bloqueio. Isso ocorre porque eles utilizam threads e simultaneidade.
- Número de threads: número de threads com os quais executar o teste.
- Tempo em segundos para executar o teste (máx = 600): o pkpspeed produz resultados medidos em “OPERAÇÕES/segundo” e relata esse valor para cada segundo que o teste é executado. Por exemplo, se o teste for executado por 5 segundos, a saída poderá ser semelhante aos seguintes valores de amostra:
 - OPERATIONS/second 821/1
 - OPERATIONS/second 833/1
 - OPERATIONS/second 845/1
 - OPERATIONS/second 835/1
 - OPERATIONS/second 837/1

Testes que podem ser executados com a ferramenta pkpspeed

- AES GCM: testa a criptografia do modo AES GCM.
- Basic 3DES CBC: testa a criptografia do modo 3DES CBC. Consulte a nota [1](#) abaixo para ver uma mudança futura.

- AES básico: testa a CBC/ECB criptografia AES.
- Digest: testa o hash digest.
- Assinar ECDSA: testa a assinatura de ECDSA.
- Verificação de ECDSA: testa a verificação de ECDSA.
- FIPS Random: testa a geração de um número aleatório compatível com FIPS (Observação: só pode ser usado no modo de bloqueio).
- HMAC: testa o HMAC.
- Random: este teste não é relevante porque estamos usando FIPS 140-2 de HSMs.
- RSA non-CRT versus RSA_CRT: testa as operações de assinatura e verificação de RSA.
- RSA OAEP Enc: testa a criptografia RSA OAEP.
- RSA OAEP Dec: testa a decodificação do RSA OAEP.
- RSA private dec non-CRT: testa a criptografia de chave privada RSA (não otimizada).
- RSA private key dec CRT: testa a criptografia de chave privada RSA (otimizada).
- Assinatura RSA PSS: testa a assinatura RSA PSS.
- Verificação RSA PSS: testa a verificação RSA PSS.
- Enc de chave pública RSA: testa a criptografia de chave pública RSA.

A criptografia de chave pública RSA, a decodificação privada RSA sem CRT e a decodificação de chave privada RSA CRT também solicitarão que o usuário responda ao seguinte:

```
Do you want to use static key [y/n]
```

Se y for inserida, uma chave pré-computada será importada para o HSM.

Se n for inserido, uma nova chave será gerada.

[1] De acordo com a orientação do NIST, isso não é permitido em clusters no modo FIPS após 2023. Para clusters no modo não FIPS, isso ainda é permitido após 2023. Para mais detalhes, consulte [Conformidade com o FIPS 140: suspensão do mecanismo de 2024](#).

Exemplos

Os exemplos a seguir mostram as opções que você pode escolher com pkpspeed (Linux) ou pkpspeed_blocking.exe (Windows) para testar o desempenho do HSM nas operações RSA e AES.

Example– Usar pkpspeed para testar a performance do RSA

Você pode executar esse exemplo no Windows, Linux e sistemas operacionais compatíveis.

Linux

Use estas instruções para Linux e sistemas operacionais compatíveis.

```
/opt/cloudhsm/bin/pkpspeed -s CU user name -p password

SDK Version: 2.03

    Available Ciphers:
        AES_128
        AES_256
        3DES
        RSA (non-CRT. modulus size can be 2048/3072)
        RSA_CRT (same as RSA)
For RSA, Exponent will be 65537

Current FIPS mode is: 00002
Enter the number of thread [1-10]: 3
Enter the cipher: RSA_CRT
Enter modulus length: 2048
Enter time duration in Secs: 60
Starting non-blocking speed test using data length of 245 bytes...
[Test duration is 60 seconds]

Do you want to use static key[y/n] (Make sure that KEK is available)?n
```

Windows

```
c:\Program Files\Amazon\CloudHSM>pkpspeed_blocking.exe -s CU user name -p password

Please select the test you want to run

RSA non-CRT----->A
RSA CRT----->B
Basic 3DES CBC----->C
Basic AES----->D
FIPS Random----->H
Random----->I
AES GCM ----->K
```

```

eXit----->X
B

Running 4 threads for 25 sec

Enter mod size(2048/3072):2048
Do you want to use Token key[y/n]n
Do you want to use static key[y/n] (Make sure that KEK is available)? n
OPERATIONS/second      821/1
OPERATIONS/second      833/1
OPERATIONS/second      845/1
OPERATIONS/second      835/1
OPERATIONS/second      837/1
OPERATIONS/second      836/1
OPERATIONS/second      837/1
OPERATIONS/second      849/1
OPERATIONS/second      841/1
OPERATIONS/second      856/1
OPERATIONS/second      841/1
OPERATIONS/second      847/1
OPERATIONS/second      838/1
OPERATIONS/second      843/1
OPERATIONS/second      852/1
OPERATIONS/second      837/

```

Example– Usar pkpspeed para testar a performance do AES

Linux

Use estas instruções para Linux e sistemas operacionais compatíveis.

```
/opt/cloudhsm/bin/pkpspeed -s <CU user name> -p <password>
```

SDK Version: 2.03

Available Ciphers:

AES_128

AES_256

3DES

RSA (non-CRT. modulus size can be 2048/3072)

RSA_CRT (same as RSA)

For RSA, Exponent will be 65537

```

Current FIPS mode is: 00000002
Enter the number of thread [1-10]: 1
Enter the cipher: AES_256
Enter the data size [1-16200]: 8192
Enter time duration in Secs: 60
Starting non-blocking speed test using data length of 8192 bytes...

```

Windows

```

c:\Program Files\Amazon\CloudHSM>pkpspeed_blocking.exe -s CU user name -p password
login as USER
Initializing Cfm2 library
    SDK Version: 2.03

    Current FIPS mode is: 00000002
    Please enter the number of threads [MAX=400] : 1
    Please enter the time in seconds to run the test [MAX=600]: 20

    Please select the test you want to run

    RSA non-CRT----->A
    RSA CRT----->B
    Basic 3DES CBC----->C
    Basic AES----->D
    FIPS Random----->H
    Random----->I
    AES GCM ----->K

    eXit----->X
D

Running 1 threads for 20 sec

Enter the key size(128/192/256):256
Enter the size of the packet in bytes[1-16200]:8192
OPERATIONS/second          9/1
OPERATIONS/second          10/1
OPERATIONS/second          11/1
OPERATIONS/second          10/1
OPERATIONS/second          10/1
OPERATIONS/second          10/...

```

AWS CloudHSM O usuário do SDK 5 do cliente contém valores inconsistentes

O `user list` comando no AWS CloudHSM Client SDK 5 retorna uma lista de todos os usuários e propriedades do usuário em seu cluster. Se alguma das propriedades de um usuário tiver o valor “inconsistente”, esse usuário não será sincronizado em seu cluster. Isso significa que o usuário existe com propriedades diferentes HSMs em diferentes no cluster. Com base em qual propriedade é inconsistente, diferentes etapas de reparo podem ser feitas.

A tabela a seguir inclui etapas para resolver inconsistências para um único usuário. Se um único usuário tiver várias inconsistências, resolva-as seguindo estas etapas de cima para baixo. Se houver vários usuários com inconsistências, analise essa lista para cada usuário, resolvendo totalmente as inconsistências desse usuário antes de passar para a próxima.

Note

Para executar essas etapas, o ideal é estar logado como administrador. Se sua conta de administrador não for consistente, siga estas etapas fazendo login como administrador e repetindo as etapas até que todas as propriedades estejam consistentes. Depois que sua conta de administrador estiver consistente, você poderá continuar usando esse administrador para sincronizar outros usuários no cluster.

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
A “função” do usuário é “inconsistente”	<pre>{ "username": "test_user", "role": "inconsistent ", "locked": "false", "mfa": [], "cluster-coverage": "full" }</pre>	Esse usuário é um CryptoUser em alguns HSMs e um administrador em outros HSMs. Isso pode acontecer se dois SDKs tentarem criar o mesmo usuário, ao mesmo tempo, com funções diferentes.	1. Faça login como administrador. 2. Exclua o usuário em todos os itens HSMs: <pre>user delete --username <user's name> -- role admin</pre>

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
	<pre>}</pre>	Você deve remover esse usuário e recriá-lo com a função desejada.	<pre>user delete --username <user's name> -- role crypto-user</pre> 3. Crie o usuário com a função desejada: <pre>user create --username <user's name> --role <desired role></pre>

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
A “cobertura do cluster” do usuário é “inconsistente”	<pre>{ "username": "test_user", "role": "crypto-user", "locked": "false", "mfa": [], "cluster-coverage": "inconsistent " }</pre>	Esse usuário existe em um subconjunto do HSMs cluster. Isso pode acontecer se for user create parcialmente bem-sucedido ou parcialmente bem-sucedido. user delete Você deve concluir sua operação anterior, criando ou removendo esse usuário do seu cluster.	Se o usuário não deve existir, siga estas etapas: 1. Faça login como administrador. 2. Execute este comando: <pre>user delete -- username<user's name> --role admin</pre> 3. Execute o seguinte comando: <pre>user delete -- username<user's name> --role crypto-user</pre> Se o usuário deve existir, siga estas etapas: 1. Faça login como administrador. 2. Execute o seguinte comando: <pre>user create --username <user's name> --role <desired role></pre>

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
O parâmetro “bloqueado” do usuário é “inconsistente” ou “verdadeiro”	<pre>{ "username": "test_user", "role": "crypto-user", "locked" : inconsistent , "mfa": [], "cluster-coverage": "full" }</pre>	Esse usuário está bloqueado em um subconjunto de HSMs. Isso pode acontecer se um usuário usar a senha errada e se conectar somente a um subconjunto HSMs do cluster. Você deve alterar as credenciais do usuário para que sejam consistentes em todo o cluster.	Se o usuário tiver o MFA ativado, siga estas etapas: 1. Faça login como administrador. 2. Execute o comando a seguir para desativar temporariamente o MFA: <pre>user change-mfa token-sign --username <user's name> --role <desired role> --disable</pre> 3. Altere a senha do usuário para que ele possa fazer login em todos HSMs: <pre>user change-password --username <user's name> --role <desired role></pre> Se o MFA estiver ativado para o usuário, siga estas etapas:

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
			1. Faça com que o usuário faça login e reative o MFA (isso exigirá que ele assine tokens e forneça sua chave pública em um arquivo PEM): user change-mfa token-sign --username <user's name> --role <desired role> --token <File>

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
O status do MFA é "inconsistente"	<pre>{ "username": "test_user", "role": "crypto-user", "locked": "false", "mfa": [{ "strategy": "token-sign", "status": "inconsistent " }], "cluster-coverage": "full" }</pre>	Esse usuário tem diferentes sinalizadores de MFA em diferentes HSMs no cluster. Isso pode acontecer se uma operação de MFA for concluída apenas em um subconjunto de HSMs Você deve redefinir a senha do usuário e permitir que ele reative o MFA.	Se o usuário tiver o MFA ativado, siga estas etapas: 1. Faça login como administrador. 2. Execute o comando a seguir para desativar temporariamente o MFA: <pre>user change-mfa token-sign --username <user's name> --role <desired role> --disable</pre> 3. Você também precisará alterar a senha do usuário para que ele possa fazer login em todos HSMs: <pre>user change-password --username <user's name> --role <desired role></pre> Se o MFA estiver ativado para o

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
			usuário, siga estas etapas: 1. Faça com que o usuário faça login e reative o MFA (isso exigirá que ele assine tokens e forneça sua chave pública em um arquivo PEM): <pre>user change- mfa token-sig n --username <user's name> --role <desired role> --token <File></pre>

AWS CloudHSM Falhas de replicação do usuário do Client SDK 5

O `user replicate` comando na CLI do CloudHSM replica um usuário entre clusters clonados do CloudHSM. Este guia aborda falhas causadas por inconsistências de usuário no cluster de origem ou entre os clusters de origem e de destino. A replicação do usuário verifica se os usuários são consistentes verificando os seguintes atributos:

- Perfil de usuário
- Status de bloqueio da conta
- Status do quórum
- Status da autenticação multifator (MFA)

Problema: o usuário selecionado não está sincronizado em todo o cluster

O processo de replicação de usuário verifica a sincronização do usuário em todo o cluster de origem. Se algum dos atributos do usuário tiver o valor “inconsistente”, isso significa que ele não está sincronizado em todo o cluster. A replicação de usuários apresenta falha com a seguinte mensagem de erro:

```
{
  "error_code": 1,
  "data": "Specified user is inconsistent across the cluster"
}
```

Para verificar a dessincronização do usuário no cluster de origem:

- Execute o comando `user list` na CloudHSM CLI.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "example-inconsistent-user",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "inconsistent"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
```

```
    "mfa": [],
    "quorum": [],
    "cluster-coverage": "full"
  }
]
```

Resolução: sincronize os atributos do usuário em todo o cluster de origem

- Para sincronizar as informações do usuário em todo o cluster de origem, consulte o seguinte: [AWS CloudHSM O usuário do SDK 5 do cliente contém valores inconsistentes](#).

Problema: o usuário existe no cluster de destino com atributos diferentes

Se um usuário já existir com a mesma referência em um ou mais HSMs no cluster de destino, mas tiver atributos de usuário diferentes, o seguinte erro poderá ocorrer:

```
{
  "error_code": 1,
  "data": "User replicate failed on 1 of 3 connections"
}
```

Resolução

1. Determine qual versão do usuário deve ser mantida.
2. Exclua o usuário indesejado usando o comando `user delete` no cluster apropriado. Consulte [Excluir um AWS CloudHSM usuário com a CLI do CloudHSM](#) para obter mais informações.
3. Replique o usuário executando o comando `user replicate`.

Problema: falha na replicação do usuário de hsm2m.medium para hsm1.medium

A replicação de usuários de hsm2m.medium para hsm1.medium não é compatível. Se replicar um usuário de um cluster de origem hsm2m.medium para um cluster de destino hsm1.medium, ocorrerá o seguinte erro:

```
{
  "error_code": 1,
  "data": "User replicate failed on 1 of 1 connections"
}
```

Resolução

- Use o [gerenciamento de usuários](#) com a CloudHSM CLI para recriar manualmente os usuários ausentes.

AWS CloudHSM Falhas de replicação de 5 chaves do SDK do cliente

O `key replicate` comando na CLI do CloudHSM replica uma chave de um cluster de origem para um cluster de AWS CloudHSM destino. AWS CloudHSM Este guia aborda falhas causadas por inconsistências no cluster de origem ou entre os clusters de origem e de destino.

Problema: a chave selecionada não está sincronizada em todo o cluster

O processo de replicação de chaves verifica a sincronização da chave em todo o cluster de origem. Se alguma das informações ou atributos da chave tiver o valor “inconsistente”, isso significa que ela não está sincronizada em todo o cluster. A replicação de chaves apresenta falha com a seguinte mensagem de erro:

```
{
  "error_code": 1,
  "data": "The selected key is not synchronized throughout the cluster"
}
```

Para verificar a dessincronização da chave no cluster de origem:

1. Execute o comando `key list` na CloudHSM CLI.
2. Use o sinalizador `--filter` para especificar a chave.
3. Adicione o sinalizador `--verbose` para ver a saída completa com as informações de cobertura de chave.

```
aws-cloudhsm > key list --filter attr.label=example-desynchronized-key-label --verbose
```



```

{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000048000f",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "example-desynchronized-key-label",
          "id": "0x",
          "check-value": "0xbe79db",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": "inconsistent",
          "trusted": false,
          "unwrap": false,
          "verify": true,
          "wrap": false,
          "wrap-with-trusted": false,

```

```
        "key-length-bytes": 16
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

Resolução: sincronize as informações e os atributos da chave em todo o cluster de origem

Para sincronizar as informações e os atributos da chave em todo o cluster de origem:

1. Para os atributos de chave inconsistentes: use o comando `key set-attribute` para definir o atributo desejado para a chave específica.
2. Para cobertura inconsistente de usuários compartilhados: use os comandos `key share` ou `key unshare` para ajustar o compartilhamento de chaves com os usuários desejados.

Problema: existe uma chave com a mesma referência no cluster de destino com diferentes informações ou atributos

Se uma chave com a mesma referência já existir no cluster de destino com informações ou atributos diferentes, o seguinte erro poderá ocorrer:

```
{
  "error_code": 1,
  "data": "Key replicate failed on 1 of 3 connections"
}
```

Resolução

1. Escolha qual versão da chave deve ser mantida.
2. Exclua a versão indesejada da chave usando o comando `key delete` no cluster apropriado.
3. Replique a chave pelo cluster que tem a versão correta.

AWS CloudHSM erro visto durante a verificação da disponibilidade da chave

Problema: Um módulo de segurança de AWS CloudHSM hardware (HSM) está retornando o seguinte erro:

```
Key <KEY HANDLE> does not meet the availability requirements - The key must be available on at least 2 HSMs before being used.
```

Causa: as verificações de disponibilidade de chaves procuram chaves que, em condições raras, mas possíveis, poderiam estar perdidas. Esse erro geralmente ocorre em clusters com apenas um HSM ou em clusters com dois HSMs durante um período em que um deles está sendo substituído. Nessas situações, as seguintes operações do cliente provavelmente causaram o erro acima:

- Uma nova chave foi gerada usando um comando como [A categoria generate-symmetric na CLI do CloudHSM](#) ou [A generate-asymmetric-pair categoria na CLI do CloudHSM](#).
- Uma [Listar chaves para um usuário com a CloudHSM CLI](#) operação foi iniciada.
- Uma nova instância do SDK foi iniciada.

Note

O OpenSSL bifurca novas instâncias do SDK com frequência.

Resolução/recomendação: escolha uma das seguintes ações para evitar que esse erro ocorra:

- Use o `--disable-key-availability-check` parâmetro para definir a disponibilidade da chave como falsa no arquivo de configuração da sua [ferramenta de configuração](#). Para obter mais informações, consulte a seção [AWS CloudHSM Parâmetros de configuração do SDK do cliente 5](#) da ferramenta de configuração.
- Se estiver usando um cluster com dois HSMs, evite usar as operações que causaram o erro, exceto durante o código de inicialização.
- Aumente a quantidade de HSMs em seu cluster para pelo menos três.

AWS CloudHSM extraindo chaves usando o JCE

Use as seções a seguir para solucionar problemas de extração de AWS CloudHSM chaves usando o JCE.

getEncoded, getPrivateExponent, ou GETs retorna null

getEncoded, getPrivateExponent e getS vão retornar nulos porque, por padrão, estão desativados. Para habilitá-los, consulte [Extração de chaves usando JCE para AWS CloudHSM](#).

Se getEncoded, getPrivateExponent e getS retornarem nulos após serem ativados, sua chave não atende aos pré-requisitos corretos. Para obter mais informações, consulte [Extração de chaves usando JCE para AWS CloudHSM](#).

getEncoded getPrivateExponent,, ou GETs retornam bytes de chave fora do HSM

Você ou alguém com acesso ao seu sistema habilitou limpar extração de chave. Consulte as páginas a seguir para obter mais informações, incluindo como redefinir essa configuração para o estado padrão desativado.

- [Extração de chaves usando JCE para AWS CloudHSM](#)
- [Proteger e extrair chaves de um HSM](#)

Controle de utilização do HSM

Quando sua carga de trabalho exceder a capacidade do módulo de segurança de hardware (HSM) do AWS CloudHSM cluster, você receberá mensagens de erro informando que HSMs está ocupado ou está limitado. Quando isso acontece, você pode ver uma taxa de transferência reduzida ou um aumento na taxa de solicitações de rejeição de HSMs. Além disso, HSMs pode enviar os seguintes erros de ocupação.

Para Client SDK 5

- Em PKCS11, os erros de ocupação são mapeados paraCKR_FUNCTION_FAILED. Esse erro pode ocorrer por vários motivos, mas se o controle de utilização do HSM causar esse erro, as seguintes linhas de registro aparecerão no seu registro:

- `[cloudhsm_provider::hsm1::hsm_connection::e2e_encryption::error]` Failed to prepare E2E response. Error: Received error response code from Server. Response Code: 187
- `[cloudhsm_pkcs11::decryption::aes_gcm]` Received error from the server. Error: This operation is already in progress. Internal error code: `0x000000BB`
- No JCE, os erros de ocupação são mapeados para `com.amazonaws.cloudhsm.jce.jni.exception.InternalException: Unexpected error with the Provider: The HSM could not queue the request for processing.`
- Outros erros de SDKs 'ocupação' imprimem a seguinte mensagem: Received error response code from Server. Response Code: 187.

Para Client SDK 3

- Em PKCS11, erros de ocupação são mapeados para `CKR_OPERATION_ACTIVE` erros.
- No JCE, os erros de ocupação são mapeados para `CFM2Exception` com o status de `0xBB` (187). Os aplicativos podem usar a função `getStatus()` em `CFM2Exception` para verificar qual status será retornado pelo HSM.
- Outros erros de SDKs ocupação imprimirão a seguinte mensagem: `HSM Error: HSM is already busy generating the keys(or random bytes) for another request.`

Resolução

É possível resolver esses problemas executando uma ou mais das seguintes ações:

- Adicione comandos de repetição para operações de HSM rejeitadas em sua camada de aplicação. Antes de ativar os comandos de repetição, verifique se o cluster está dimensionado adequadamente para atender às cargas máximas.

Note

Para o Client SDK 5.8.0 e versões posteriores, os comandos de repetição são ativados por padrão. Para obter detalhes sobre a configuração do comando de repetição de cada SDK, consulte [Configurações avançadas para a ferramenta de configuração do Client SDK 5](#).

- Adicione mais HSMs ao seu cluster seguindo as instruções em [Escalabilidade HSMs em um cluster AWS CloudHSM](#).

 Important

Recomendamos testar a carga do seu cluster para determinar a carga máxima que você deve prever e, em seguida, adicionar mais um HSM a ele para garantir a alta disponibilidade.

Mantenha os usuários do HSM sincronizados HSMs em todo o cluster AWS CloudHSM

Para [gerenciar os usuários do seu HSM](#), você usa uma ferramenta de linha de AWS CloudHSM comando conhecida como `cloudhsm_mgmt_util`. Ele se comunica somente com os HSMs que estão no arquivo de configuração da ferramenta. Ele não conhece outros HSMs no cluster que não estejam no arquivo de configuração.

AWS CloudHSM sincroniza as suas chaves com HSMs todas as outras HSMs no cluster, mas não sincroniza os usuários ou as políticas do HSM. Quando você usa `cloudhsm_mgmt_util` para [gerenciar usuários do HSM](#), essas alterações de usuário podem afetar somente alguns dos clusters HSMs — aqueles que estão no arquivo de configuração `cloudhsm_mgmt_util`. Isso pode causar problemas ao AWS CloudHSM sincronizar chaves HSMs no cluster, porque os usuários que possuem as chaves podem não existir em todas HSMs no cluster.

Para evitar esses problemas, edite o arquivo de configuração `cloudhsm_mgmt_util` antes de gerenciar os usuários. Para obter mais informações, consulte [???](#).

Conexão perdida com o AWS CloudHSM cluster

Ao [configurar o AWS CloudHSM cliente](#), você forneceu o endereço IP do primeiro HSM em seu cluster. Esse endereço IP é salvo no arquivo de configuração do AWS CloudHSM cliente. Quando o cliente é iniciado, ele tenta se conectar a esse endereço IP. Se não puder, por exemplo, porque o HSM falhou ou foi excluído, os seguintes erros podem aparecer:

```
LIQUIDSECURITY: Daemon socket connection error
```

```
LIQUIDSECURITY: Invalid Operation
```

Para resolver esses erros, atualize o arquivo de configuração com o endereço IP de um HSM ativo e acessível no cluster.

Para atualizar o arquivo de configuração do AWS CloudHSM cliente

1. Use uma das seguintes maneiras para encontrar o endereço IP de um HSM ativo no cluster.
 - Veja a HSMguia na página de detalhes do cluster no [AWS CloudHSM console](#).
 - Use o AWS Command Line Interface (AWS CLI) para emitir o [describe-clusters](#) comando.

Esse endereço IP será necessário em uma etapa subsequente.

2. Use o comando a seguir para parar o cliente.

Amazon Linux

```
$ sudo stop cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

CentOS 8

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client stop
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client stop
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe stop AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

Use Ctrl + C na janela de comando em que você iniciou o AWS CloudHSM cliente.

3. Use o comando a seguir para atualizar o arquivo de configuração do cliente, fornecendo o endereço IP que tiver encontrado em uma etapa anterior.

```
$ sudo /opt/cloudhsm/bin/configure -a <IP address>
```

4. Use o comando a seguir para iniciar o cliente.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```


CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```

RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Para clientes Windows 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Para o cliente Windows 1.1.1 e anterior:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe  
C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Registros AWS CloudHSM de auditoria ausentes CloudWatch

Se você criou um AWS CloudHSM cluster antes de 20 de janeiro de 2018, precisará configurar manualmente uma [função vinculada ao serviço](#) para permitir a entrega dos registros de auditoria desse cluster. Para obter instruções sobre como habilitar uma função vinculada ao serviço em um cluster do HSM, consulte [Noções básicas sobre funções vinculadas ao serviço](#), bem como [Criar uma função vinculada ao serviço](#) no Guia do usuário do IAM.

Personalizado IVs com comprimento não compatível para embalagem de chave AES AWS CloudHSM

Este tópico de solução de problemas ajuda a determinar se seu aplicativo gera chaves agrupadas irre recuperáveis. Se você for afetado por esse problema, use este tópico para resolvê-lo.

Tópicos

- [Determine se seu código gera chaves agrupadas irre recuperáveis](#)
- [Ações que você deve realizar se seu código gerar chaves agrupadas irre recuperáveis](#)

Determine se seu código gera chaves agrupadas irre recuperáveis

Você será afetado somente se atender a todas as condições abaixo:

Condição	Como posso saber?
Seu aplicativo usa a biblioteca PKCS #11	A biblioteca PKCS #11 está instalada como o arquivo <code>libpkcs11.so</code> na sua pasta <code>/opt/cloudhsm/lib</code> . Aplicativos escritos na linguagem C geralmente usam a bibliotec a PKCS #11 diretamente, enquanto aplicativ os escritos em Java podem estar usando a biblioteca indiretamente por meio de uma camada de abstração Java. Se você estiver usando o Windows, NÃO será afetado, pois a biblioteca PKCS #11 não está disponível atualmente para Windows.

Condição	Como posso saber?
Seu aplicativo usa especificamente a versão 3.0.0 da biblioteca PKCS #11	Se você recebeu um e-mail da AWS CloudHSM equipe, provavelmente está usando a versão 3.0.0 da biblioteca PKCS #11. Para verificar a versão do software nas instâncias do seu aplicativo, use este comando: <pre>rpm -qa grep ^cloudhsm</pre>
Você agrupa as chaves usando o agrupamento de chaves AES	Agrupamento de chave AES significa que você usa uma chave AES para agrupar alguma outra chave. O nome do mecanismo correspondente é CKM_AES_KEY_WRAP . Ele é usado com a função C_WrapKey . Outros mecanismos de empacotamento baseados em AES que usam vetores de inicialização (IVs), como CKM_AES_GCM e CKM_CLOUDHSM_AES_GCM , não são afetados por esse problema. Saiba mais sobre funções e mecanismos.
Você especifica um IV personalizado ao chamar o agrupamento de chaves AES, e o comprimento desse IV é menor que 8	O agrupamento de chave AES geralmente é inicializado usando uma estrutura CK_MECHANISM da seguinte forma: <pre>CK_MECHANISM mech = {CKM_AES_KEY_WRAP, IV_POINTER, IV_LENGTH};</pre> Isso se aplicará somente se for definido como: • IV_POINTER não é NULL • IV_LENGTH é menor que 8 bytes

Se não atender a todas as condições acima, você pode parar de ler agora. Suas chaves agrupadas podem ser desagrupadas adequadamente e esse problema não afeta você. Caso contrário,

consulte [the section called “Ações que você deve realizar se seu código gerar chaves agrupadas irre recuperáveis”](#).

Ações que você deve realizar se seu código gerar chaves agrupadas irre recuperáveis

Você deve seguir as três etapas a seguir:

1. Atualize imediatamente sua biblioteca PKCS #11 para uma versão mais recente
 - [Biblioteca PKCS #11 mais recente para Amazon Linux, CentOS 6 e RHEL 6](#)
 - [Biblioteca PKCS #11 mais recente para Amazon Linux 2, CentOS 7 e RHEL 7](#)
 - [Biblioteca PKCS #11 mais recente para Ubuntu 16.04 LTS](#)
2. Atualize seu software para usar um IV compatível com os padrões

É altamente recomendável que você siga nosso código de exemplo e simplesmente especifique um NULL IV, o que faz com que o HSM utilize o IV padrão compatível com os padrões. Como alternativa, é possível especificar explicitamente o IV como 0xA6A6A6A6A6A6A6A6 com um comprimento IV correspondente de 8. Não recomendamos o uso de nenhum outro IV para agrupamento de chaves AES e desativaremos explicitamente a personalização IVs para empacotamento de chaves AES em uma versão futura da biblioteca PKCS #11.

O código de amostra para especificar corretamente o IV aparece em [aes_wrapping.c](#) em GitHub

3. Identifique e recupere chaves agrupadas existentes

Você deve identificar todas as chaves que você empacotou usando a versão 3.0.0 da biblioteca PKCS #11 e, em seguida, contatar o suporte para obter assistência (<https://aws.amazon.com/support>) na recuperação dessas chaves.

Important

Esse problema afeta apenas as chaves agrupadas com a versão 3.0.0 da biblioteca PKCS #11. É possível agrupar chaves usando versões anteriores (2.0.4 e pacotes com numeração inferior) ou versões posteriores (3.0.1 e pacotes com numeração superior) da biblioteca PKCS #11.

Resolvendo falhas na criação de AWS CloudHSM clusters

Quando você cria um cluster, AWS CloudHSM cria a função vinculada ao serviço do AWSService RoleForCloud HSM, se a função ainda não existir. Se você AWS CloudHSM não conseguir criar a função vinculada ao serviço, sua tentativa de criar um cluster poderá falhar.

Este tópico explica como resolver os problemas mais comuns para que você possa criar um cluster com sucesso. Você precisa criar essa função apenas uma vez. Uma vez que a função vinculada a serviço for criada na sua conta, você poderá usar qualquer um dos métodos com suporte para criar clusters adicionais e gerenciá-los.

As seções a seguir oferecem sugestões para solucionar falhas de criação de cluster relacionadas à função vinculada a serviço. Se você tentou, mas ainda não conseguiu criar um cluster, entre em contato com a [Suporte](#). Para obter mais informações sobre a função vinculada ao serviço do AWSService RoleForCloud HSM, consulte [Funções vinculadas a serviços para AWS CloudHSM](#)

Tópicos

- [Adicionar a permissão ausente](#)
- [Criar a função vinculada ao serviço manualmente](#)
- [Usar um usuário não federado](#)

Adicionar a permissão ausente

Para criar uma função vinculada a serviço, o usuário deve ter a permissão `iam:CreateServiceLinkedRole`. Se o usuário do IAM que está criando o cluster não tiver essa permissão, o processo de criação do cluster falhará ao tentar criar a função vinculada ao serviço em sua AWS conta.

Quando uma permissão ausente causa a falha, a mensagem de erro inclui o seguinte texto.

This operation requires that the caller have permission to call `iam:CreateServiceLinkedRole` to create the CloudHSM Service Linked Role.

Para resolver esse erro, forneça ao usuário do IAM que está criando o cluster a permissão `AdministratorAccess` ou adicione a permissão `iam:CreateServiceLinkedRole` à política do IAM desse usuário. Para obter instruções, consulte [Adicionar permissões a um usuário novo ou existente](#).

Em seguida, tente [criar o cluster](#) novamente.

Criar a função vinculada ao serviço manualmente

Você pode usar o console, a CLI ou a API do IAM para criar a função vinculada ao serviço do AWSService RoleForCloud HSM. Para obter mais informações, consulte [Criar uma função vinculada ao serviço](#) no Guia do usuário do IAM.

Usar um usuário não federado

Usuários federados, cujas credenciais são originárias de fora do AWS, podem realizar muitas das tarefas de um usuário não federado. No entanto, a AWS não permite que os usuários façam chamadas de API para criar uma função vinculada a serviço a partir de um endpoint federado.

Para resolver esse problema, [crie um usuário não federado](#) com a permissão `iam:CreateServiceLinkedRole` ou dê a permissão `iam:CreateServiceLinkedRole` a um usuário não federado existente. Em seguida, peça a esse usuário que [crie um cluster](#) na AWS CLI. Isso cria a função vinculada a serviço na sua conta.

Depois de criada a função vinculada a serviço, se você preferir, poderá excluir o cluster que o usuário não federado criou. A exclusão do cluster não afeta a função. Depois disso, qualquer usuário com as permissões necessárias, incluindo usuários federados, pode criar AWS CloudHSM clusters em sua conta.

Para verificar se a função foi criada, abra o console do IAM em <https://console.aws.amazon.com/iam/> e escolha Roles. Ou use o comando [get-role](#) do IAM na AWS CLI.

```
$ aws iam get-role --role-name AWSServiceRoleForCloudHSM
{
  "Role": {
    "Description": "Role for CloudHSM service operations",
    role policy statement
    "RoleId": "AR0AJ4I6WN5QVGG5G7CBY",
    "CreateDate": "2017-12-19T20:53:12Z",
    "RoleName": "AWSServiceRoleForCloudHSM",
    "Path": "/aws-service-role/cloudhsm.amazonaws.com/",
    "Arn": "arn:aws:iam::111122223333:role/aws-service-role/
cloudhsm.amazonaws.com/AWSServiceRoleForCloudHSM"
  }
}
```

Recuperando registros de configuração AWS CloudHSM do cliente

AWS CloudHSM oferece ferramentas para o Client SDK 3 e o Client SDK 5 para coletar informações sobre seu ambiente para que o AWS Support solucione problemas.

Tópicos

- [AWS CloudHSM Ferramenta de suporte do Client SDK 5](#)
- [AWS CloudHSM Ferramenta de suporte do Client SDK 3](#)

AWS CloudHSM Ferramenta de suporte do Client SDK 5

O script do AWS CloudHSM Client SDK 5 extrai as seguintes informações:

- O arquivo de configuração do componente Client SDK 5
- Arquivos de log disponíveis
- A versão atual do sistema operacional
- Informações do pacote:

Executar a ferramenta de informações para o Client SDK 5

O Client SDK 5 inclui uma ferramenta de suporte ao cliente para cada componente, mas todas as ferramentas funcionam da mesma forma. Executa a ferramenta para criar um arquivo de saída com todas as informações coletadas.

As ferramentas usam uma sintaxe como esta:

```
[ pkcs11 | dyn | jce | ksp | cli ]_info
```

Por exemplo, para coletar informações para suporte de um host Linux executando a biblioteca PKCS #11 e fazer com que o sistema grave no diretório padrão, você executaria este comando:

```
/opt/cloudhsm/bin/pkcs11_info
```

A ferramenta cria o arquivo de saída dentro do diretório `/tmp`.

AWS CloudHSM CLI

Para coletar dados de suporte para AWS CloudHSM CLI no Linux

- Use a ferramenta de suporte para coletar dados.

```
/opt/cloudhsm/bin/cli_info
```

Para coletar dados de suporte para AWS CloudHSM o Windows

- Use a ferramenta de suporte para coletar dados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cli_info.exe"
```

PKCS #11 library

Para coletar dados de suporte para a biblioteca PKCS #11 no Linux

- Use a ferramenta de suporte para coletar dados.

```
/opt/cloudhsm/bin/pkcs11_info
```

Para coletar dados de suporte para a biblioteca PKCS #11 no Windows

- Use a ferramenta de suporte para coletar dados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\pkcs11_info.exe"
```

OpenSSL Dynamic Engine

Para coletar dados de suporte para o OpenSSL Dynamic Engine no Linux

- Use a ferramenta de suporte para coletar dados.

```
/opt/cloudhsm/bin/dyn_info
```


JCE provider

Para coletar dados de suporte para o provedor JCE no Linux

- Use a ferramenta de suporte para coletar dados.

```
/opt/cloudhsm/bin/jce_info
```

Para coletar dados de suporte para o provedor JCE no Windows

- Use a ferramenta de suporte para coletar dados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\jce_info.exe"
```

Key Storage Provider

Para coletar dados de suporte para o Provedor de armazenamento de chaves no Windows

- Use a ferramenta de suporte para coletar dados.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\ksp_info.exe"
```

Recuperar logs de um ambiente de tecnologia sem servidor

Para configurar ambientes sem servidor, como Fargate ou Lambda, recomendamos que você configure seu tipo de log como. AWS CloudHSM term Uma vez configurado paraterm, o ambiente sem servidor será capaz de produzir para. CloudWatch

Para obter os registros do cliente CloudWatch, consulte [Trabalho com grupos de registros e fluxos de registros](#) no Guia do usuário do Amazon CloudWatch Logs.

AWS CloudHSM Ferramenta de suporte do Client SDK 3

O script do AWS CloudHSM Client SDK 3 extrai as seguintes informações:

- Sistema operacional e sua versão atual
- Informações sobre a configuração do cliente dos arquivos `cloudhsm_client.cfg`, `cloudhsm_mgmt_util.cfg` e `application.cfg`

- Registros do cliente a partir do local específico da plataforma
- Informações de cluster e HSM usando `cloudhsm_mgmt_util`
- Informações sobre OpenSSL
- Versão atual do cliente e da compilação
- Versão do instalador

Executar a ferramenta de informações para o Client SDK 3

O script cria um arquivo de saída com todas as informações coletadas. O roteiro cria o arquivo de saída dentro do diretório `/tmp`.

Linux: `/opt/cloudhsm/bin/client_info`

Windows: `C:\Program Files\Amazon\CloudHSM\client_info`

Warning

Esse script tem um problema conhecido nas versões 3.1.0 a 3.3.1 do SDK do Cliente 3. É altamente recomendável que você atualize para a versão 3.3.2, que inclui uma correção para esse problema. Consulte a página [Problemas conhecidos](#) para obter mais informações antes de usar essa ferramenta.

AWS CloudHSM cotas

As cotas, anteriormente conhecidas como limites, são os valores atribuídos aos AWS recursos. As cotas a seguir se aplicam aos seus AWS CloudHSM recursos por AWS região e AWS conta. A cota padrão é o valor inicial aplicado por AWS, e esses valores estão listados na tabela abaixo. Uma cota ajustável pode ser aumentada acima da cota padrão.

Cotas de serviço

Recurso	Cota padrão	Ajustável?
Clusters	4	Sim
HSMs	6	Sim
HSMs por cluster	28	Não

A maneira recomendada de solicitar um aumento de cota é abrir o [console do Service Quotas](#). No console, escolha seu serviço e cota e envie sua solicitação. Para obter mais informações, consulte a [documentação do Service Quotas](#).

As quotas na tabela Cotas do sistema a seguir não são ajustáveis.

Cotas do sistema

Recurso	Cota para hsm1.medium	Cota para hsm2.medium
Máximo de chaves por cluster	3.300	Total de 16.666 chaves, com no máximo 3.333 chaves assimétricas
Máximo de usuários por cluster	250	1,024
Comprimento máximo do nome do usuário	31 caracteres	31 caracteres
Comprimento obrigatório da senha	De 8 a 32 caracteres	De 8 a 32 caracteres

Recurso	Cota para hsm1.medium	Cota para hsm2.medium
Número máximo de conexões de clientes simultâneas por cluster ¹	900	900
Número máximo de sessões PKCS #11 por aplicativo	1,024	1,024

[1] Uma conexão de cliente para o Client SDK 3 é um daemon de cliente. Para o Client SDK 5, uma conexão de cliente é um aplicativo.

Downloads para o SDK AWS CloudHSM do cliente

Os tópicos a seguir fornecem downloads para o AWS CloudHSM cliente SDKs.

Note

Para obter informações sobre quais plataformas são suportadas por cada Client SDK, consulte [AWS CloudHSM Plataformas compatíveis com o Client SDK 5](#) e [AWS CloudHSM Plataformas compatíveis com o Client SDK 3](#).

Tópicos

- [AWS CloudHSM versão mais recente do SDK do cliente](#)
- [AWS CloudHSM versões anteriores do Client SDK](#)
- [AWS CloudHSM versões obsoletas do Client SDK](#)
- [AWS CloudHSM end-of-life Lançamentos do SDK do cliente](#)

AWS CloudHSM versão mais recente do SDK do cliente

Em março de 2021, AWS CloudHSM lançou o Client SDK versão 5.0.0, que apresenta um SDK de cliente totalmente novo com diferentes requisitos, recursos e suporte de plataforma.

O Client SDK 5 é totalmente compatível com ambientes de produção e oferece os mesmos componentes e nível de suporte do Client SDK 3. Para obter mais informações, consulte [Compare AWS CloudHSM o suporte a componentes do SDK do cliente](#).

Note

A partir do SDK 5.17 do CloudHSM, estamos atualizando nosso contrato de nível de serviço para oferecer suporte a até três versões secundárias anteriores e um ano a partir da data de lançamento de uma versão do SDK. Lançaremos novas versões do SDK do CloudHSM para oferecer novos recursos e para correções de segurança, estabilidade e desempenho. Desativaremos os links de download para versões mais antigas e sem suporte do SDK do CloudHSM à medida que novas versões forem disponibilizadas.

Esta seção inclui a versão mais recente do Client SDK. Para obter informações sobre versões anteriores, consulte [Versões anteriores do SDK](#). Para versões obsoletas, consulte [Versões obsoletas do SDK](#).

Versão 5 do SDK do cliente: versão 5.17.0

Amazon Linux 2023

Baixe o software versão 5.17.0 para Amazon Linux 2023 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum e0656d606c1f75b8ecf11df79f2d4aa17c0b68983bb20d271dabe02f7b97ca47)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum b8917abe48799035018bc044fbc2e9db8d5568ebf29663ecc12d53ec2be5efd2)
- [Provedor OpenSSL](#) (SHA256 checksum 61e49cb0f1f9eb7bcf76f5ef36f01cf1fadfa56bf5eab94b05efdd93214c9cf7)
- [Provedor JCE](#) (SHA256 checksum 4ba2548fa27f3662c56fc140386366cfb40cde899695412c6d4731a7d0a70e43)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 0a9de06b3f15d38f3aa02dd0288d3a9ad41f525f692f539664168b9e6b95e18d)

Baixe o software versão 5.17.0 para Amazon Linux 2023 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum 9bd40e13bf777a4651721ebe5613cf2e4f0f339679d50f06b81f7f1b73c1fef5)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum e592419bc0449664f0707281a46ebb775091386ac5b297e35ff0e4160468cbe0)
- [Provedor OpenSSL](#) (SHA256 checksum 46358ef68388928702d780564811814aa826c143ccc0624a2a45d8995226d33e)
- [Provedor JCE](#) (SHA256 checksum db5bd1494c742a64ca3f2bc57253b17655ad38ecc27d45422965466c1caf740f)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum e3d21b4460a421058f65a597b0bb30a9fdda890363ebee107b124a1212d5a042)

Amazon Linux 2

Baixe o software versão 5.17.0 para Amazon Linux 2 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 17d96d5f4ad25de74fadcc7272fac825d43c496b0b7912e4d32b1ccfb67250bc)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 3d3de61bd16ef849e53827709b8c8bd9eaf89bd1b7d56b1a02455bca38fe1eb6)
- [Provedor JCE](#) (SHA256 checksum 5c268313ebf16cf762619a1e75f1d44efdbd6eaf5ba4bbce67ecfe96bbbed9775)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 43b678eff0225590cc59c02645332f4459c8161a6742e562eb95902d1bec9134)

Baixe o software versão 5.17.0 para Amazon Linux 2 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum 7910ec55f8b106d804290b77e6d13cefb85ef79cbc728f69bb21eebeb7245c6b)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 399c16e0fbfa372710a83836517cfeaae0895513931427a458e7994ce91a321d)
- [Provedor JCE](#) (SHA256 checksum 1678581e31fcfe58625ca1e3c0762c4b96095065a868a5b7f0872214e5ab9e8)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 52257b580d73d902e9fd29f2c334f88cabee484c7eec8af28f7b2aa7864466e7)

RHEL 10 (10.0+)

Baixe o software da versão 5.17.0 para RHEL 10 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 037102ee2b7f9c244d647cfb06da282b99349e208d27b7b0aa80bd32e382a2d4)
- [Provedor OpenSSL](#) (SHA256 checksum 3d6d97df03a003bae579dabbf9abeefc1bd2fa6d799a6db01c3ef98d02145393)
- [Provedor JCE](#) (SHA256 checksum 4e69ec89acfefc19f922c68988d4674718d49d5cda2e0a288f7f0343ebfc270c)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 38a67191949cc8cb0c69ef2fe4694514b2ca6af4cf5f8763bd7f6860c501675a)

Baixe o software da versão 5.17.0 para RHEL 10 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum bc383aff236d4b2a0fdbc5fd81907cf9212ad8f3cfc91fadedb6cdc14b9d3d581)
- [Provedor OpenSSL](#) (SHA256 checksum d9ead84f8d2e6a09044646d62fed293ca082431e5e0338bcbbf8acac023ecc0)
- [Provedor JCE](#) (SHA256 checksum c61fdb5f05cf486619deb3e23ecb171ce4485bb64c891bf4480f6e6a4276c3bc)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum fbd4387ef3fab177de8c888d98d1b24bc042d23ea2f93834dfb5ee918518f406)

RHEL 9 (9.2+)

Baixe o software da versão 5.17.0 para RHEL 9 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 7fc68025493fc4a0bc4d3b2fd11b17da4481b6bee51fc86a40f95fe4c206aa60)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
3e786e9218f657c93e1bf1b54c81f0f573e39d10147cf99485da73677b0d67ae)
- [Provedor OpenSSL](#) (SHA256 checksum c52b366f9c6c960d697794bd7840888bc51e1ee5aa654b8d2c83ec1626797c56)
- [Provedor JCE](#) (SHA256 checksum 427e9c68815188b59a0f9b2fd14793791d9a4ce4d3e5ddc34fce916af18a45f1)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 49d128639eea0e588c95281635a33b0e62837cf5a9d992e0bacb5b598a0037ed)

Baixe o software da versão 5.17.0 para RHEL 9 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum
3ae8f5068abb3d9ca75ac21b160acc7ee84c7ded6107a889602448af51d5bd82)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
64c5feaf86fc40e28616156f17942cc46d8ac93babdd8af59dd98a1f2e73117d)
- [Provedor OpenSSL](#) (SHA256 checksum 24df4296ba0ae15f6b7060ccffeca0d1a24f58d9e8da8eb6a4afa3a6a50770fd)
- [Provedor JCE](#) (SHA256 checksum 648d7948ab8a1271d7cf0eddc2ca32e960979be481b99a1ce2ce8b9fa3a2b2ed)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum edd1f3707aeea285a55cf24d2e2385489458a69e5d7dab4014783f5cdc232d77)

RHEL 8 (8.3+)

Baixe o software da versão 5.17.0 para RHEL 8 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
85e070ed3273ead9698711dfc2c3e590dda1a01f99074943459ddc5ac052872d)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
d0c8e5603f4c99d0ba547c0ba40361c15977f3349ac120d514afed59abf32c83)
- [Provedor JCE](#) (SHA256 checksum 4567b9a236d0dfd0410863beaced6d5d4e71acb06b2b136c7c40ce6e704f3698)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum f8a090752849fac9bf99ef67b59f571e476908dccec1285b466c25e6cc71793c)

Baixe o software da versão 5.17.0 para RHEL 8 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum
71ade413435947eb2093727cb0e7a5c5f084ca859cb38d728a8c3d23f3b49bd0)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
02210da06b3fd22813fce8b89b753d150c8f321df405c40dd9f0172dc03968a7)
- [Provedor JCE](#) (SHA256 checksum 80ac5dbaad1f644d6b9551e6a1d95d4b4a51ae82d34e45593be77b1068f7c910)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum c3c56c3b9fed1dc51761d9412007c727d3b608377a089bd637a44bc374744d5c)

Ubuntu 24.04 LTS

Baixe o software da versão 5.17.0 para o Ubuntu 24.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum b3869a6fe9f2031cb5f36ac8d2a5a5280451159e7a68f0d6f91f98b5f0100939)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
3e46443a3f8425f62cec8d8ca414c4b5446b86b42012877ebe44a17303403413)
- [Provedor OpenSSL](#) (SHA256 checksum 69c2e61894e954ac1d0a785025902af5aa1922a7f6011a847f2111e153db706e)
- [Provedor JCE](#) (SHA256 checksum 663da1a7b13a1055bb90fd7b168b69fcfb000eb91c1c3ce6062d8cc8db607fe4)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 0d57b4f08bd768ed595978d2a6d3fe6bbbf8398a784c66c4e30c19ecf6730b5d)

Baixe o software da versão 5.17.0 para o Ubuntu 24.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum
94d9a36a3d3f2f04c10539713b98ddad68654c739572064e4c123005a9caf575)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum a442950055a7051f50b4cdfa7ec347c1b207b0bbf7e159b31a3bf96896b66146)
- [Provedor OpenSSL](#) (SHA256 checksum dcc4b5a370da176a86e47683a78703a5604d6b7b49ba687377460f48c3ddff26)
- [Provedor JCE](#) (SHA256 checksum 1fe1fa59a3e5c32edd5668604e43f7ce152473514b25ee736dfb0e9cf87c92c9)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 2bf2c13bf1520eaf2a45eb55aaba3334fa54da686d22d1a03005ec8f62ea8895)

Ubuntu 22.04 LTS

Baixe o software versão 5.17.0 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 2dc1f00952f69b59b0498e85c2f2009f80c9a2f9f28104995b0291da5f45d805)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum df0e7ff8852a93fb4c5807317d3438ab96ffe69f19ba144d872991cbcb4172f6)
- [Provedor JCE](#) (SHA256 checksum 77a3cc37e066ae0ad5655ddc3dd38a0513bdcdbd3c9d3d780e2973ac42a75035e)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 88ca1614178771991649b8117ddf27ffb99c6842ee89f68e77c6df3470e6ba78)

Baixe o software da versão 5.17.0 para o Ubuntu 22.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 0e1f62783c16debf9188c9cf0085fe972671e5ea9bf0fc2fc3255f908885949c)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 109d25583e2d5ed0fe821a77098d8f2c815011f7873168e0110aeb857dc903ab)
- [Provedor JCE](#) (SHA256 checksum 9151661a39f6394aa3ace682bd7964095ed5300244582d3f6b1f81255ebf22cf)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 66e408cea800a93becf378b8611595ad8cd1bf51e0ae7feebc548fe3099b477c)

Windows Server 2025

Baixe o software versão 5.17.0 para Windows Server 2025 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- [Provedor JCE](#) (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Windows Server 2022

Baixe o software da versão 5.17.0 para Windows Server 2022 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- [Provedor JCE](#) (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Windows Server 2019

Baixe o software versão 5.17.0 para Windows Server 2019 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- [Provedor JCE](#) (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Windows Server 2016

Baixe o software versão 5.17.0 para Windows Server 2016 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- [Provedor JCE](#) (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- [CloudHSM CLI](#) (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum 3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

O Client SDK 5.17.0 apresenta novos recursos e melhorias em vários componentes.

Suporte à plataforma

- Foi adicionado suporte ao RHEL 8 para ARM64 arquitetura.

JCE

- Foi adicionado suporte para o OpenJDK 2.5 em todas as plataformas
- O Client SDK 5.17.0 é a última versão compatível com o OpenJDK 8.

Novos atributos

- Introduziu o novo OpenSSL Provider SDK para melhorar a compatibilidade com o OpenSSL 3.2+ e operações criptográficas adicionais.
- Foi adicionado suporte à curva Ed25519 à geração de chaves EC.
- Foi adicionado suporte para assinatura e verificação do EdDSA (Edwards-curve Digital Signature Algorithm) na CLI do CloudHSM.

CloudHSM CLI

- Foi adicionado suporte à curva Ed25519 à geração de chaves EC para tipos hsm2m.medium não FIPS.

- Foi adicionado suporte para assinatura e verificação do EdDSA (Edwards-curve Digital Signature Algorithm) na CLI do CloudHSM. Para obter mais informações, consulte [Gere uma assinatura com o mecanismo HashEd DSA na CLI do CloudHSM](#) e [Verifique uma assinatura assinada com o mecanismo HashEd DSA na CLI do CloudHSM](#).

Provedor OpenSSL

- O novo OpenSSL Provider SDK fornece integração com a arquitetura OpenSSL 3.2+ Provider para tipos hsm2m.medium.

Correções de bugs e melhorias:

- Mecanismos de repetição aprimorados para operações criptográficas.

AWS CloudHSM versões anteriores do Client SDK

Esta seção lista as versões anteriores do Client SDK.

Versão 5.16.2

Amazon Linux 2023

Baixe o software versão 5.16.2 para Amazon Linux 2023 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
e223649cbf689afaeca445b962cceaab3c143b901ee7974a7a5360b6424c409b)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
9b51e8b1429f22705399299c111385e9434873a1dd26dda2c70498646590148c)
- [Provedor JCE](#) (SHA256 checksum 45ec10c2e3d592fb1fe56cc9dee1d8653e3a6dfa15b8185618fd3cf66c567314)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 8134a0e316286310eda1606b6a14e1c9fd6f0e27934a43e55e24d3094ef870a6)

Baixe o software versão 5.16.2 para Amazon Linux 2023 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum
7556c69294d0c67b359cf9995ec7047cdc90e21542d2c9d5e110e35f13c329c5)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 6ab4f3b9731a6f26891658f5e3a0ff74255096390c42aa7f702ba07e0bd924b1)
- [Provedor JCE](#) (SHA256 checksum 98b80696762a73f3f9be6c3c7af0f8f9db23ba63d66d4d941690b6698019702)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 649397606f527f330049cf7adb5ad667126dfd7f63f878f1f46b469e2f127814)

Amazon Linux 2

Baixe o software versão 5.16.2 para Amazon Linux 2 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 073ee736ad9c7de6e6451afdecc4d77e6ed1a63f314a7fc09373ff35986fa0fc)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum b174fb24a15f1e22dd520ea9ac4ac14229c2adce14ac6df8901f9d3c0e7e24f6)
- [Provedor JCE](#) (SHA256 checksum f18e42f2679ab0fb51c421965a5dd6c8fe1361442d92403d1d230bedb003b27a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 5a6c83833b60017dd494c31fd4abf3896f57177be7951674c85d0d4e909d2041)

Baixe o software versão 5.16.2 para Amazon Linux 2 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum df2c97a27855916983f242eba05e0741335158432ac46ebbeae47c4a3b7b0bef)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum d25456e3da39c63d9c0d4a928815f995239b146f4f0b26a8c50c742905543207)
- [Provedor JCE](#) (SHA256 checksum 7e31734efbbae49203e57d7ad4d1e291faa5b7e16c469a3596da82a7380bceeb)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum b77eb1f9ff58ccc864854d7f261e331ce50836cd770efc8e8eaebb33d78c23e2)

RHEL 10 (10.0+)

Baixe o software da versão 5.16.2 para RHEL 10 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 8044cb40c9c65ae944900c740308e68eeb0156d3b93051d60463cd3f6095ef67)
- [Provedor JCE](#) (SHA256 checksum a1b7f110c2b15ba7ce9adee74baf1f108d85146269321991314f9153a074ca27)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 3ded29fdb74537b284cf871af91dd64fce8483770141a3e4470e94fb69524621)

Baixe o software da versão 5.16.2 para RHEL 10 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 643fa23ffea47dfb21fd326cacf99e0e45f44d767ca6198ab06023239eda5d2c)
- [Provedor JCE](#) (SHA256 checksum e6b5c2760c60f5c82a248d36f3b47cea9b1bbb743cf1c02a1c13554b70ab87f8)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum a6af281ff9f429f964ac9d2babc251b4423748d2b2e6da22fc51b9fd5c59714c)

RHEL 9 (9.2+)

Baixe o software da versão 5.16.2 para RHEL 9 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum e0a44849c9cb562cf803ff182e54a35e7eee682efa2a99fdfac51a996822d126)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 97463a8a88239d4a81854f2a4da7e0371dbcb375a662c7eae7753c371ef3c39f)
- [Provedor JCE](#) (SHA256 checksum 10cd4b9826451b7e9a11390cabcc7df548f5b3197e8bc87f5efe04605e503dc1)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 6c7fc59dcce4d7bc0c4460ae7c793f13fcbb306924942bc6e1457f729602f76f)

Baixe o software da versão 5.16.2 para RHEL 9 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 7b8d3f5e2bd09f30ddc1599619c63aa4452b16e1037bf10af070c055ae48cbd7)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 4edb0315c050b3b0699cb244fe36b987c9fd2921999cf7519fb669354d2efe2d)
- [Provedor JCE](#) (SHA256 checksum eae2064d9523df9f819ef4c7950a0ad15ef8bb01d8dcec062ee2558f5bebf029)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum f9f4073ebcde2b654b706efb1c0f99c08fe24903576e760029097b0e3c783f6a)

RHEL 8 (8.3+)

Baixe o software da versão 5.16.2 para RHEL 8 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 9a3e023fd5ac8c444ef07bb28b96e81b467e066c9dc2da19f648f38eb04d05c9)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 6fc6ca58f16fe2f486f4730cd0a6029eeebae98b3aed41fd8c605fe5c36c7a0)
- [Provedor JCE](#) (SHA256 checksum cb6e9eccfeebca7377b1fb815c2ae306b39b271513ce73a6cd5f1dcc55eac4c)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum e848f3f2221e7f6b984bd2001331f9e90f20e6d7351484e4e2f12431c00d87e5)

Ubuntu 24.04 LTS

Baixe o software versão 5.16.2 para o Ubuntu 24.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum d993f8c84d15cf93432b73cb37693c4ab68b3e765c35742adef3f947744b9893)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 055f0e4cc74ec7ead77fff3970422b4f1729e26a2cf3cc2be530f0e6a06f887e)
- [Provedor JCE](#) (SHA256 checksum 5bd62f8f5cb3ff4ae9430020aa4f5c8a29cb635674d2400d989f512c370c755c)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum de263f6035cf7280a982434448aa079164050e4a09508816fd06bf000cf15f0c)

Baixe o software da versão 5.16.2 para o Ubuntu 24.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum a6496f28f82f3e92b0acc8e3ae1fbe230a6d3ed82586e8ca6fe6d2f82c275e7c)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 82275d6ac2628b4291162e3609d6788b4cf01d6488a9b6a84ed8787a5a96e873)
- [Provedor JCE](#) (SHA256 checksum 67800957c69efca0fe01623364cb3b152f2780d15624955130d507a3f591e8e5)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 088f53d75250aa0ce3db7a7ba3f06ad7804b3dbeed6226683a3252890a867ac6)

Ubuntu 22.04 LTS

Baixe o software versão 5.16.2 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum b9799ba75fb8885782283abae7f49aa76a72d27f692654dd3a415157f0ea9043)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 3c7675e9b7fa4bb692865559f09fc1f9fd78dadb97d8bd6b5d1fc03367d75b86)
- [Provedor JCE](#) (SHA256 checksum 37c21821ad4d9fde9551112e8a6d7ec6b7bb62a540c0c08b6edd650d5c87b5cf)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 7d3a724892e993aebfb06ad5c3d1ef8fc650d65dff0a7de3cb9c17c089badc58)

Baixe o software versão 5.16.2 para o Ubuntu 22.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 8a2b392e48b7f9021966f6c457444ddd0964f9bb85b7ecb988bc0d372bbef6c9)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum cca0736d208d59d070ced8b82181e24919f21fbf0497e5a8a2f40436e06a8dd9)
- [Provedor JCE](#) (SHA256 checksum c12e1b9bfe1cc78ecf35892050dd06904c6a29736cc628ac1481dca4ec3e3bfe)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum 586ee7663d54e76652e9f073772fcbda218dc3c6432a221d664ffbabee3590cd)

Windows Server 2025

Baixe o software versão 5.16.2 para Windows Server 2025 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- [Provedor JCE](#) (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)

- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Windows Server 2022

Baixe o software versão 5.16.2 para Windows Server 2022 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- [Provedor JCE](#) (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Windows Server 2019

Baixe o software versão 5.16.2 para Windows Server 2019 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- [Provedor JCE](#) (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Windows Server 2016

Baixe o software versão 5.16.2 para Windows Server 2016 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- [Provedor JCE](#) (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- [CloudHSM CLI](#) (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)

- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

O Client SDK 5.16.2 oferece suporte à plataforma RHEL 10 e remove o suporte ao Ubuntu 20.04 LTS. O Client SDK 5.16.2 também adiciona suporte ao JCE Provider e ao KSP com novos recursos, ao mesmo tempo em que oferece correções de bugs e melhorias em outros. SDKs

Suporte à plataforma

- Suporte adicionado ao RHEL 10 para as arquiteturas x86 e ARM.
 - O OpenSSL Engine não é compatível no RHEL 10 como parte da transição da Red Hat para a nova arquitetura baseada em provedores introduzida no OpenSSL 3.0.

CloudHSM CLI

- Corrigido um erro em que a assinatura com dados pré-hash não limpava os tokens no modo interativo.
- Corrigido um bug em `quorum token-sign list` que causava erros anteriormente quando três ou mais tokens estavam presentes.

Provedor JCE

- Foi adicionado suporte para encontrar chaves de curva elíptica (EC) usando `ECPParameters` e `ECPoint` para tipos `hsm2m.medium`.
- Inclusão de suporte ao parâmetro `sharedInfo` para ECDH com X9.63 KDF.

Biblioteca PKCS #11

- Foi adicionado suporte para encontrar chaves de curva elíptica (EC) usando `ECPParameters` e `ECPoint` para tipos `hsm2m.medium`.

Correções de bugs e melhorias:

- Funcionalidade aprimorada de listagem de tokens para exibir mais tokens em uma única operação.
- Correção do problema de eliminação do token de uso da chave no modo interativo da CloudHSM CLI ao assinar com dados pré-hash.

- Resolução de um problema conhecido em que algumas operações limitadas do HSM em hsm2m.medium não eram repetidas automaticamente. Para mais detalhes, consulte [Problema: as operações podem falhar durante a criação do backup](#).

Versão 5.16.1

Amazon Linux 2023

Baixe o software versão 5.16.1 para Amazon Linux 2023 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 342e81846436708cfc1fb459a7ef1c7b065d8a68b50a5e63653c24918d0338a0)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 4e83bebcd20201c04629a03b70136df8d225d3056346789054c4e2ce8c9b3cba)
- [Provedor JCE](#) (SHA256 checksum 014e3c804a56087f8855ed30e480f9c193def0e1132a4769eae3ecd235d76c4f)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 854c5c21f9fbef4f53a710340f213c272365c2bb1233106266eca5a5aa547e11)

Baixe o software versão 5.16.1 para Amazon Linux 2023 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum dbe9498d91e1e2e8f80fc870a4a6576b0dc801dc20cb7a82bfdded79b3a362ac0)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 76cff941f36275163db146e05bf5fe64440248643ceb5dfcb3b64103f4f016a7)
- [Provedor JCE](#) (SHA256 checksum 8c38082797172b5630c74a8fc0c9e2652f0898bd2232b72911fbbaaa0260b5a6)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 2a0640e32405a7db138afb2444a542eab11c0a7d9610d5406478c7ac4602a14b)

Amazon Linux 2

Baixe o software versão 5.16.1 para Amazon Linux 2 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum cfa59458ec239553c86ac55773bebc69a5dc7cdc08a3927917fd5918e10abe93)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 49e50bece7d73f3b7dc95eed1df856a6dee40e27f24f40f015c6a4a2e8dee839)
- [Provedor JCE](#) (SHA256 checksum 85c8853860aa36a6c54e75c94d607a334d64dd34683ba70430ff5be61eedcc56)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 0efc3783e8429331aa446e45776fef46c7f9726a4768763ba4881dc2b05e090e)

Baixe o software versão 5.16.1 para Amazon Linux 2 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum dad60a0380ef0ea9c469cae4de10dc47124bbecedafb6956d009734cd49abceb)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 7ee303421d94544cbe9df03022c48af327833e631d8f1ec59a466673b6e9395d)
- [Provedor JCE](#) (SHA256 checksum 5ebe8157abb042ee92a7edfbfcb98bf79fc3a9907684565176fd28c387c08e88)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 5e76b4a9021a3c92e59fc608c8263af731013835738977f376fb8ad9189add56)

RHEL 9 (9.2+)

Baixe o software da versão 5.16.1 para RHEL 9 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 696e58f74d89bd8e39f85bddc547f8c94fa2fbca498318a7bb94f22e8be7a668)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 86197cd8bdd70db91331bb8380ea094352b4087c95a04768d2cefc3bba18dffa)
- [Provedor JCE](#) (SHA256 checksum 37ccc2df3e8aadda74dc060f7f5bbe63e4769311f2ccbf313a8ab5d8831f206)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 45a6db22db9ae64c239b49ea9281d29d2b4364dd63ee2408f463de17c4da1877)

Baixe o software da versão 5.16.1 para RHEL 9 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum fd79fbccf504f53a6e70497135fcea4e20982fca75b628124c7510dcbaddf56c)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 23fe7407be9c5f8da0ae64e560a9741887bc31936f0b88c8d1490e3c6893a8a8)

- [Provedor JCE](#) (SHA256 checksum bd1c43e3e28c2f71e78a41342c064ff873fd8e052f89f90227fe00eaa28f089e)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 3e877a232303052b8a6b2b869dcf228edbb0da913d3d41393f622831f4455a27)

RHEL 8 (8.3+)

Baixe o software da versão 5.16.1 para RHEL 8 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum d396c53c229b1eaa7cb30d4fcd17addc9170c7942795d1c82b23a157ec379686)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum c01f2cb66a6c5be839906b25c6a7f7990507b1b8ec3da34c128c1a90838e48df)
- [Provedor JCE](#) (SHA256 checksum 1745ab78de0712111659e91301f80e771e1b7ebc2ff8c4cc5e37bce591058520)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum b3ca40ba66062856ef63848c71f6e9dfa0a46a2b18d44c2d96a798fc4a4fb9cf)

Ubuntu 24.04 LTS

Baixe o software versão 5.16.1 para o Ubuntu 24.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum ade113ee72547cb6a8fe91a4f3ac403f462d0acd0a306f01d16ac90699f9b59c)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 9e00cb32b16fbb286a853f65c5c5154c3e805ad67820d829a0c5343f12cc9e00)
- [Provedor JCE](#) (SHA256 checksum 4fab322a138e14372aba394de29f444af640479dd338539be84e0cf659c8993b)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum f8c1e369885eb77ba4517388dab47c067a617e860fb67a028a1ebde0f96acaeef)

Baixe o software da versão 5.16.1 para o Ubuntu 24.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 5e15a69f8fb1429ed5c10429c91323fe82a63c0fe7bcc2f968820733d1737449)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum f5136ee61bd34b74d59a01d37964b70a0f80ec5981f732b265af1c5466309e6f)

- [Provedor JCE](#) (SHA256 checksum acf5bebe7009461cdc1e31b95caadcbf80da09ec78d8f69142072da9baf61b79)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 511376d2faf9f991f2193d08f80fc1edd53bf06daaa6fd8bd7a3fba0d6563ad1)

Ubuntu 22.04 LTS

Baixe o software versão 5.16.1 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum e1d787f10ee51d94732732811e4d2110f1b8e448e67fd47df8b53a2f7e56e3c8)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 29a9c41379754ce098a025feadc026f2f75a8638981f5b95ed07007d5b3d8510)
- [Provedor JCE](#) (SHA256 checksum 390d21e499ba181dc2774c3140433f6e60b49d9de3a892a89038c27e1aff157b)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum cadadafaa9464f944e5c2446df92fb1777a9e1e5116cab1b359a8aa101cf7ae8)

Baixe o software da versão 5.16.1 para o Ubuntu 22.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 3b6a3d50b4a55c150f12308ee3914451897bc4cc4c8420accf1d2d706316fcce)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum c366a3c5faf4de32a6dedc4613234ba3d331b0abac3241bed7d25e0109a44f64)
- [Provedor JCE](#) (SHA256 checksum 705416b47bc62781b5a6dfc78dd02dad4f18f842b569c3e210b9535813b9e70)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 811b501615cf34665b70f103905094ca84ed7c126f72d91b040aacda99dbf22b)

Ubuntu 20.04 LTS

Baixe o software versão 5.16.1 para o Ubuntu 20.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 6a2c33c78ada33fb435bbf8939b7cde3efe968e9f03250083dc6024b7ebd45b8)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 6ee178454e78d88e0ab92cd7c5b056d0c04cd6de192aea731d7ebcbd4c5ed761)
- [Provedor JCE](#) (SHA256 checksum 1e61ccd8d2b37c64467051d26e8d8bd592465ecef04282d6f4a9491707ba059d)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum 645693f2ede4fa0d7879eedaa41ae23120d902f24c9a637ba088f277e703cb96)

Windows Server 2025

Baixe o software versão 5.16.1 para Windows Server 2025 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)
- [Provedor JCE](#) (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Windows Server 2022

Baixe o software versão 5.16.1 para Windows Server 2022 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)
- [Provedor JCE](#) (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Windows Server 2019

Baixe o software versão 5.16.1 para Windows Server 2019 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)

- [Provedor JCE](#) (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum 80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Windows Server 2016

Baixe o software versão 5.16.1 para Windows Server 2016 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)
- [Provedor JCE](#) (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- [CloudHSM CLI](#) (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum 80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

O Client SDK 5.16.1 adiciona suporte para assinatura e verificação de dados pré-hash na CloudHSM CLI.

Suporte à plataforma

- O SDK 5.16.1 é a última versão a oferecer suporte à plataforma Ubuntu 20.04 LTS. Para obter mais informações, consulte o [site do Ubuntu](#).

CloudHSM CLI

- Inclusão de suporte para assinatura e verificação de dados pré-hash na CloudHSM CLI. Para obter mais informações, consulte [A categoria de assinatura criptográfica na CLI do CloudHSM](#) e [A categoria de verificação de criptografia na CLI do CloudHSM](#).

Provedor JCE

- Modo de AES/CBC/Pkcs5Padding criptografia atualizado para gerar automaticamente um vetor de inicialização (IV) aleatório quando IV nulo é fornecido. Anteriormente, o IV nulo resultaria em uma falha na operação. Os explícitos IVs permanecem obrigatórios para operações de descriptografia.

Correções de bugs e melhorias:

- Latência reduzida para operações repetidas de atualização de resumo no JCE.
- Atualização do comando “generate key-reference” para nomear corretamente os arquivos de referência de chave KSP com base em seus valores de ID de atributo quando executados no Windows Server. Para obter mais informações, consulte [Gerar referências de chave KSP \(Windows\)](#).

Versão 5.16.0

Amazon Linux 2023

Baixe o software versão 5.16.0 para Amazon Linux 2023 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum d63271304f32f49838390a58d94a2140ae8a744ac64efcf0e6a65983c858d862)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 25d78c1df82355601ed6887bb47d64a06380d001da15a070cfa89dce65417fb6)
- [Provedor JCE](#) (SHA256 checksum fcd876813e5d114e2b042765ff98b256410e988a4e373ea1aae6f260aebdd554)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 7f1441eb08daa7ceab91161900d89974ee86e0a3c54d878a29e43db775bd04f5)

Baixe o software versão 5.16.0 para Amazon Linux 2023 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum 0b42e4e89cddda10c5be5f295a52d7a541bba057a3a07d3d9b192cf9eb49c776)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum e8ee949d44b9f9ba64ed36eb7944acacbd1d76d7cf09d5492daee5928a9953e9)
- [Provedor JCE](#) (SHA256 checksum 67a6fd8f7b5dfc60ec6d6d380c31035675373b542205ec7e5a687175191cd275)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 1e6c3053b19aded54045a3baa9fe2545f81a35332366cf2d77dca92ee888e654)

Amazon Linux 2

Baixe o software versão 5.16.0 para Amazon Linux 2 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 5a89ff9801f89f51e27e70869f2713cfe9bbd87d1198246cfaafe74aff9e809)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
991b6288289d07972915aa0be6dfd6b3c33c1d6312bc304225715e30832e688f)
- [Provedor JCE](#) (SHA256 checksum c5e673f79c2efd83195a288a5c8860c4bf74dba5bc8e422741dd5cc1be230fb3)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum b9021b0b348c08f433e7cce247ab1710e2a79f33a82ac36e1d6a4672171028db)

Baixe o software versão 5.16.0 para Amazon Linux 2 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum
97677a7f9759637b548205d31c9a08a3e7b27dfa5a38e9a4e8e2398ae9869a5a)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
dde369dc4efd9f59e5c9a5459a532ad5d273f87839fba6dd5dcde2c3a2b61517)
- [Provedor JCE](#) (SHA256 checksum b7d96e7453e26ff07ccd9437b102f4c21857c79daa6315054728fc7ac66150bc)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum c7386ac016b14ea8cb96a6ed655a67f46826cdb925bd87902277b5ce9b593d51)

RHEL 9 (9.2+)

Baixe o software da versão 5.16.0 para RHEL 9 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 178d3e1ff0ec8cdc7f8c8be9aab772b4195602bb1c48dea692b628689be5ea3d)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
0488ad66cd825fatee70d1a1ccb1c8045b089af7b7d6ccadbec75f535376048c)
- [Provedor JCE](#) (SHA256 checksum cc9a60fa981a46c674de1e715d92ec1b8cd5c801394bfb3f0d101880589d4202)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum cb7e3ce6704f015d5b9462f12e6ad5489366fb3e80f554a2bdf7f04e688be1a9)

Baixe o software da versão 5.16.0 para RHEL 9 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 9567fb016b641821869919d4d6e9754ae08fd549381446640ff135a1e225b809)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
8c46a9dd908e77a0d8412e7050621d0c570e091845480815c230e26d1917227b)
- [Provedor JCE](#) (SHA256 checksum 78a7c714aedd068e5e2f12f5044a6ce1618e6dfacbccdd91b4a9b505842dcc6c9)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum d2340ef4092ca6c1fa1ac8b484550508a5d9d8670e663bef6e595a07f8151be3)

RHEL 8 (8.3+)

Baixe o software da versão 5.16.0 para RHEL 8 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
47bf628a4a2663b8e8c7207b43e5af1d05ed2247e85e1371773596c132c9a4ec)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
49dd44243863121f7781e0f4ac556828c746d85db723864ccf16ea5fb042846a)
- [Provedor JCE](#) (SHA256 checksum c5e8a6664e6a8e5338a98b3845bed029f5936b517e6043aeb65ad58c3364eed8)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 71f2cd6bd8892cca8a845bfe575cdf20016dd7176e5216b76290e499c822a2f2)

Ubuntu 24.04 LTS

Baixe o software versão 5.16.0 para o Ubuntu 24.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum fd7e78050aba017b7dc859f0b1e6f354f56b22091cb262b328aad204064a2960)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
9d2a9e8049caa12c8027ad2963fa30f22e1032d4025e39a4e2d5985073c283f5)
- [Provedor JCE](#) (SHA256 checksum ae19c13f55e01538e989e8693aaf3a5d998c60f1ed71e721a34404a228947c20)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 7ebedb3e8c4b8e01cc1e5dfe32ae18c9542d9c2940c77bd71ee102093aaf1d1c)

Baixe o software da versão 5.16.0 para o Ubuntu 24.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum
cfd9526cf495e81a2477b464ca3c124a89d07b1a366a14b39eeb27d6b5e2404b)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
02a5cc2cd9442b7bf982bb3ae44c9b134758b46e68e46fe4729b4adf1320d75c)
- [Provedor JCE](#) (SHA256 checksum 124d04f667a7222d624f528404a289405669f4e4a1983b8a7926758bf67e35ec)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum b874556b34ff5f0d082ca19f62822cf97d6e7074ce3861d7577090f99cba0cbd)

Ubuntu 22.04 LTS

Baixe o software versão 5.16.0 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
e36d9bf900e195e417db261034d820cb935bec5135a0aa332d46019c70257cac)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
2c39129a5acce96b8693ce8c4b4aa174d9c13bc8ea3c2652fa505b4e0a933842)
- [Provedor JCE](#) (SHA256 checksum 03b70d834985a2500b73eecd831bff4f30389eb67460bf1d799466e59a9f9cfd)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 3fc4fc662188ebbd430d9b5e01a57774567b926f3478d0a455fee4968c47460d)

Baixe o software versão 5.16.0 para o Ubuntu 22.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum c835d9bb604b5c47a702fd45267ffb9dfa37d68dd0734391f060835ea680a169)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
943213a0f14b50fbdf3a90286e7f7ab219ab77750698f5062013873419484a04)
- [Provedor JCE](#) (SHA256 checksum 0ecc3e65a3b94c2dd3913f44aba58764e5558a8bedd3b742a42e4769533f3734)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 8558328b224b850cf9e1b3c4f8773565a90e38ab43e2c9e90bc5b841f98898d2)

Ubuntu 20.04 LTS

Baixe o software versão 5.16.0 para o Ubuntu 20.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 64a01e84d898aca3cd03f22022cf8dbbc806a1735a84df5f820ba5bfa3339b0e)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
53a73cfc60c3fddd3ad9173bf6e602faf289f20935b3c469a137847148442954)
- [Provedor JCE](#) (SHA256 checksum 09544cb8129af4d06a241e1fbc0b6e6c207e1a518e5cb202f7b6c9802507d1b8)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum e1fdf1e014541f57d3ba4688bad88e7557e32ba974e020b4da76f85f1fc6aa29)

Windows Server 2025

Baixe o software versão 5.16.0 para Windows Server 2025 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- [Provedor JCE](#) (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Windows Server 2022

Baixe o software versão 5.16.0 para Windows Server 2022 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- [Provedor JCE](#) (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Windows Server 2019

Baixe o software versão 5.16.0 para Windows Server 2019 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- [Provedor JCE](#) (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Windows Server 2016

Baixe o software versão 5.16.0 para Windows Server 2016 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- [Provedor JCE](#) (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- [CloudHSM CLI](#) (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

O Client SDK 5.16 adiciona ECDH com suporte a X963 KDF no tipo de cluster hsm2m.medium para o provedor JCE e a biblioteca PKCS #11. O Client SDK 5.16 também adiciona suporte para gerar arquivos de referência de chave KSP no Windows Server com a CloudHSM CLI.

CloudHSM CLI

- Inclusão de suporte para gerar arquivos de referência de chave KSP no Windows Server; para obter mais informações, consulte [Gerar referências de chave KSP \(Windows\)](#).

Provedor JCE

- Inclusão de ECDH compatível com X963 KDF no tipo de cluster hsm2m.medium, consulte [Mecanismos compatíveis com o provedor JCE para o AWS CloudHSM Client SDK 5](#).

Biblioteca PKCS #11

- Inclusão de ECDH compatível com X963 KDF no tipo de cluster hsm2m.medium, consulte [Mecanismos compatíveis com a biblioteca PKCS #11 para o AWS CloudHSM Client SDK 5](#).

Correções de bugs e melhorias:

- Correção de um erro em que as chaves de sessão não eram limpas adequadamente durante a desconexão.
- Correção de um problema em que operações bem-sucedidas de logout do mTLS em hsm2m.medium retornavam incorretamente uma resposta de erro.
- Correção de um problema com falha de conexão durante a inicialização do Windows.
- Correção de um problema em que as ferramentas de informações do SDK exibiam uma saída incorreta nos sistemas Ubuntu.
- Correção de um problema que anteriormente permitia ECDSA com hashes mais fracos no modo FIPS.

Versão 5.15.0

Amazon Linux 2023

Baixe o software versão 5.15.0 para Amazon Linux 2023 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 41ef3178811df1dbb03b2cbac83fe0f4768bdc9b17005c409f1c0229f93ef11c)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum afa1f9f8bd99f54866dea1b8928c00b951a6e492f5f36d0d6c7c38fff341d609)

- [Provedor JCE](#) (SHA256 checksum 6ea775e05570ef3497a4df5c35a6ec1c682aea73c48e7fec3e541af995759e)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 78c10bb213dd14fcfc5836e358de6aedac61db05125fd61137b0082214fdecbe)

Baixe o software versão 5.15.0 para Amazon Linux 2023 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum cd0617d29b6d64c00c8e14fd9a92f604e14acda4746f4f0b86b9de42367192fb)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum bc9acfd04eb1246eb3d5b0a8f3736ec017c0d1699d5395f85868d4a1722cd83)
- [Provedor JCE](#) (SHA256 checksum 96d05301486206577b7be05aac561649167b57272c9c06ff839fd8ff2b5d96d5)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum b870eadb27736a2cde98022d57e9704c67ae15878cf0b910738859cdabaa35a2)

Amazon Linux 2

Baixe o software versão 5.15.0 para Amazon Linux 2 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum c70ae4f0181a8187c9380481c51c1d03e12236dd86863ec818ed3f210b294c8e)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 08e9fd1dd80efa637f9a1727bb0de205ba124a3776b2e8bc21008ee458063a42)
- [Provedor JCE](#) (SHA256 checksum 7932ed060e72c53b2556f30694b0ffe5342b244b6628c7a9dc03966aa49c8fe6)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 0a939e4d5d0a2ff308c7a1d9e73ebc865e426214d556bde1bd29dbe807fbb583)

Baixe o software versão 5.15.0 para Amazon Linux 2 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum bcf907a8fb4722b09c54b8e5785fe2b8cfcdd6ee3fdda2d21879a012138077)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 8f70edc3a6a4a1bf0264c6567b1dda1ac69055f206753f88eeadbb8bf3bf9f38)
- [Provedor JCE](#) (SHA256 checksum 75dd67736bb08fe7e46e113af10803a255ba8edee3016ca963c1ee94fe59d43b)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 1c2ecf90c955281d99bcd8d1956d63debb15bbc8419744c83f88821ef8b78aee)

RHEL 9 (9.2+)

Baixe o software da versão 5.15.0 para RHEL 9 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 65bd0b815eebc806674a7bf7c54e9f884595881547f5fffd08ff6a38aabdcce)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
f2af9f5882ab2e5a11defecc660f8af5c4d9d6e2e2b89873e6833fc2976f44ac)
- [Provedor JCE](#) (SHA256 checksum 5124bace5d1544775c891f13a0e309b30dd73699116c46ecc9a77bba5f9cf633)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 1eec31fb2c0ab3c2839d9bdb37874cf8dbc74a383279068fce9e8613966d06e0)

Baixe o software da versão 5.15.0 para RHEL 9 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum
f498697519afa04b6b0362a4388a6a38e2cb6813b781d6ce3d97a7de89c5cbfe)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
5410da63108a1b209e567e9bccf8bd7e4035af88b0d58b9d78b10917be1b40c1)
- [Provedor JCE](#) (SHA256 checksum b46f233e6994d2c0ed505dc5c717ee3009daaeb2063d260aa78e06273770bffa)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum e9f93eaa58db2f7ea1174164ef96ab219700933d353243c3c6ab1aebac5ccffe)

RHEL 8 (8.3+)

Baixe o software da versão 5.15.0 para RHEL 8 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
87131e179d0e60ade302ec07b22803cfb39294bf060b786c41f154d95791ac94)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
f412a2f5cd761db5940288bb252ce060d44735c6b436bb6d4fa7d3687a44a026)

- [Provedor JCE](#) (SHA256 checksum d0844f55c08f9ff9c393138a9041efe1b59dc3dce20c0b2c23406efe6acc43db)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 9fd8033e478ce6d7d640c063c4f007359cb04c19d519826a745ad0885f96a0f8)

Ubuntu 24.04 LTS

Baixe o software versão 5.15.0 para Ubuntu 24.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum ca5f2f80ae921cfebd5c8bc35c39d2b19cfabfd5981932409eaf2e7c00a9097)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum e44cd7b678a421957c84e4fc0f70280360fd4e1e66f4cabdd1b20b955ee5fcca)
- [Provedor JCE](#) (SHA256 checksum bc6382f12769c3d87c34522d0d616b7a8c108574c003814e1300938c386655ac)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum f786be86c204680e670f7817376ff376733ebd96b51c31e9c98213078596637f)

Baixe o software da versão 5.15.0 para o Ubuntu 24.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 1ecf057d67137c0e9e1bb0dda57d581690c647b0360c3a617a8dc668919d08de)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 4f3d23e6b798f88be587108d8e6a225d796a3d080aef61ea384eb74a1270612a)
- [Provedor JCE](#) (SHA256 checksum b9fd16bdcc1fcf59fde0d3e0debee500b0b7edfdff69209e84d14393097fe9d2)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum a95922d9b44e64a64723db0c21ac89566515a5a6c87de990af4a9e1f40c7424f)

Ubuntu 22.04 LTS

Baixe o software versão 5.15.0 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 966be12eb32de813ca07e766abf7b5616c0d2e105e9296d920aadaca10e5afdf)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum f2840151d87b7f9cbff68993c25397afd48a16f054abf0f2fd4624662d3087d6)
- [Provedor JCE](#) (SHA256 checksum 01aebdda05640e50a82cae04c5cb6d33ab909dadd917bd834957d4f75ad8c577)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 2e5eba9409abc429828505779c512cd2424719766c0e488f50aee288966cf61d)

Baixe o software da versão 5.15.0 para o Ubuntu 22.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 614f74f101c9c64fff515128c5a59fa23de5047494fe248e8aabdb441092b3e)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum c15d6db77b76bce690749b73b947567eb5f2d76669887843116d0ce56c1f8ea7)
- [Provedor JCE](#) (SHA256 checksum bca8511d5c0a173b0fef326016ce5091b6e6829fa2b3cb45ed5621290ae3e42a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 2f5b673148b682d7e34619c51b5e8799abe7dc7fd4f046158a0d05320ba24dc1)

Ubuntu 20.04 LTS

Baixe o software versão 5.15.0 para o Ubuntu 20.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 86c8394b5ddff91a71194fb87c327efde36baa2380e559c04f9d543a6e74d61b)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 0a4227389fea61e6e7ac7cfa715eb341f7a4eeae9ed10e4c96da2c0dd4a18f9e)
- [Provedor JCE](#) (SHA256 checksum 65b2d926ff9dfbe6c7864bc3a41b3da2383bc731dd199bcef8805a0543fbe612)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 09c2e55bcf72f9e530717950d8c5fd48574ae6ccb09a049526ca5b2a3b8aa9)

Windows Server 2025

Baixe o software versão 5.15.0 para Windows Server 2025 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- [Provedor JCE](#) (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)

- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Windows Server 2022

Baixe o software versão 5.15.0 para Windows Server 2022 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- [Provedor JCE](#) (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Windows Server 2019

Baixe o software versão 5.15.0 para Windows Server 2019 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- [Provedor JCE](#) (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum
ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum
52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Windows Server 2016

Baixe o software versão 5.15.0 para Windows Server 2016 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)

- [Provedor JCE](#) (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
- [Javadocs para AWS CloudHSM](#) (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- [CloudHSM CLI](#) (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum 52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

O Client SDK 5.15 adiciona suporte para replicar um usuário em clusters clonados com a CloudHSM CLI. O Client SDK 5.15 também adiciona pacotes de instalação para a biblioteca PKCS #11, o provedor JCE, a CloudHSM CLI e o Key Storage Provider (KSP) para Windows Server 2025.

Suporte à plataforma

- Inclusão de suporte ao Windows Server 2025 para a biblioteca PKCS #11, provedor JCE, CloudHSM CLI e provedor de armazenamento de chaves (KSP).

CloudHSM CLI

- O seguinte comando foi incluído:
 - Replicar um usuário com a CloudHSM CLI

Versão 5.14.0

Amazon Linux 2023

Baixe o software versão 5.14.0 para Amazon Linux 2023 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 05e7a3882166c694a7a09bc735f08f91c8145a4215176665eacacdf3e509abe8)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum f4dd9966988418e100c276dc0d521f91afdfc0e6c008dbf8eda446ebaca83c14)
- [Provedor JCE](#) (SHA256 checksum c4dee5c1173f6a1c7683aedb58e61d329c36933435c416f288d94bc9a68a6b31)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 1fad069fde305450254287e43750d597db9af0cb8fd168300cd5eaed9e2af33a)

Baixe o software versão 5.14.0 para Amazon Linux 2023 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum 9f34163d02bce26c1280e589310cda891d27995c50cec0a7fe083100ecff2b69)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 7444a4daad6e4715c82d6c39a7b03a07ee0201a13fe0f98da96acbf9d24abf6c)
- [Provedor JCE](#) (SHA256 checksum 3393fe3a0f5c3a9c92106d74b7de074c818e095f97d2cfd600dbd47779b90b37)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum d9a2edce48c5f6646d5a351cb431712f2d2fc62d21f8318e7bb1ce579819d7f4)

Amazon Linux 2

Baixe o software versão 5.14.0 para Amazon Linux 2 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum 9c47b90bfa0ad51627cdb0dd8f148a56090fbdeb2490f1ab4009170c7b9c1120)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 215f9768331565085a317585b3dbe0514b251fdc428c96ed32491c4abb9fea56)
- [Provedor JCE](#) (SHA256 checksum a802f941e95fcbf0ef37775fc096c0d6ae4c916fa08330a9d56defc5f99ff2b7)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum ac10f23dd81264b1d10a0760f62b5006d2b2b66bf1c6378248ca9326afb65a83)

Baixe o software versão 5.14.0 para Amazon Linux 2 na ARM64 arquitetura:

- [Biblioteca PKCS #11](#) (SHA256 checksum f2d1550f043c655ef5d51e2ca14a0416886d99902e0702bd7f30f93b2c563d4d)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 2c8e2c81af53ba3646d1f947894d84b9b54780bbca79b58f354125e4ac9427c0)
- [Provedor JCE](#) (SHA256 checksum 3f9c881056e6905d46358585db72143b59971958a708fa4ac75cb53994487213)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 6d5985e44c9852409dd3d342239fbbf7d0f9ad43d445e0980153c9b9eafe2b6f)

RHEL 9 (9.2+)

Baixe o software versão 5.14.0 para RHEL 9 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum c4812210421ff2fa5dac8477a2e6b10552aabd88b1f03d717e044e7293823158)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 3f6aeaa6ae1faae7d8bba1596f358cad1eec9e562cf08aaab9ded92cabe94719)
- [Provedor JCE](#) (SHA256 checksum bd8f120b08f738ad4d1534b0e32aad903758d5e75d3aba7cb9ff6a77dec533db)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 84f13a9babd767edf90dabb0b14034ec5b2208898123a13966dd6b8519961c27)

Baixe o software da versão 5.14.0 para RHEL 9 em arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 3faa1de563aa9773939cab7b39907dcb7b981ed7225019f9070a9dd52db7ae70)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 13b41ac47ef7ee7bf78f585b8347ca4da9ebc296e4fc1e6a0c2ff5b333354ca6)
- [Provedor JCE](#) (SHA256 checksum 06803655ebe54d59c180bb17ac6fe56337bdd58ad0f6fe87c50ff8df32f70258)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum e6b4e9688d0db9d72bbee3450fe19736d640c9931adbd4f4ef73cb7ac2a08cf4)

RHEL 8 (8.3+)

Baixe o software da versão 5.14.0 para RHEL 8 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum e400aeaa6dbf7721f97e71c643a0db4f5f1094fb197fc46dc0ab293de9d16f2d)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum bfcc27d251e62f9eba0fd508e7d08dc62126642d4cdd0b5566183957768b8c54)
- [Provedor JCE](#) (SHA256 checksum b5500031b572c918a8df4a0347e01c8ea00a7366b865b310bd92427fa1ed53e3)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 000712d0a691efc64a6c5d54bfeb1ab48b315ebb5dd6926b0502e326bf700291)

Ubuntu 24.04 LTS

Baixe o software versão 5.14.0 para Ubuntu 24.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
bbec70a198a4b173620b4018acc297ce6a6a80d372706e0101997d6bca35bca)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
fcb77f75bd465b22401a09a20c410985833340295101263b7171cdcc4ac9f980)
- [Provedor JCE](#) (SHA256 checksum e1fa16aae2f6095c89a8bf392ef2cb9ca4db8853c858904ff90abf4bb491d74b)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum af32e0ac1c5f8c7f8cb40d255abe7c63ce9d981293187a3fb452562fa05756f4)

Baixe o software versão 5.14.0 para Ubuntu 24.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 1232318b084347889f92136536537ca519373683ce39bcec20f6ac3fa2f42d7c)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
1745ee3a33d8e6ea72644e903c55c6a206204cf0a8bea200bc4a7b15736ed801)
- [Provedor JCE](#) (SHA256 checksum 7a44acabbc90c996594ed53661937f9242850823347f7c386f02fc041a97471a)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum e6778bd12c55fd152b50033833531fe569472f4f2bd9927a345eb126e8305739)

Ubuntu 22.04 LTS

Baixe o software versão 5.14.0 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum
6b4b1620e9a85267950633b171dd188b7ac7094e371e188fab1bef7a911a16f)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum
fee5f0a65fab0f46ad58689af5dc510721581f31364d3be5cbbf79f5d9a60db8)
- [Provedor JCE](#) (SHA256 checksum 5883a3d15e160d65f8e26f185e1ee30f68becad5f6fcd16abc1c4586689800b5)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum
173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 7b1a8cd2962f4be7325154c080dabfb820beac5629e272c3a3ebc0e6cab11d27)

Baixe o software versão 5.14.0 para o Ubuntu 22.04 LTS na arquitetura: ARM64

- [Biblioteca PKCS #11](#) (SHA256 checksum 8680f63fd74272ea0e4d4c32dfb96b3eb9b60e03483c202d4e9b65ee101a178f)

- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum c757304e8fc5f38be3bab7ac6d37a35dcb56d31e62f7194da62b3a176593d1d8)
- [Provedor JCE](#) (SHA256 checksum a78ee59341da56af315d126ec7ed9f0dafa6e99649f659db2436be3863cb035b)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 18931bfff869a0bd54846b3296d870fa19beba5652f979352be8fba6307e6d1aa)

Ubuntu 20.04 LTS

Baixe o software versão 5.14.0 para o Ubuntu 20.04 LTS na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum f1461c16b135ebcc17deec46aab88bd113ea122b8942fc188d4f05cd03e919a8)
- [Mecanismo dinâmico do OpenSSL](#) (SHA256 checksum 89211a7a7ed50eda2dc385c31ea76f1fbabd389ca691204873531d983c3eb0f7)
- [Provedor JCE](#) (SHA256 checksum f098c32d61a53b073459a75d88b68e377a9b16335874fae060cb10df0da00df0)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum ff91fb930717c917344af2ba344dc6e02bd5abc004dcb6147e9412b67e2aa7ab)

Windows Server 2022

Baixe o software versão 5.14.0 para Windows Server 2022 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum ee5a8e8e85fff7acd0bbafa23740e3981f7dc52e708972b600c2b26603786838)
- [Provedor JCE](#) (SHA256 checksum 2ae0274f09f66981c03fd1e3c264e896ba7cd211168ea31369335db1b3ea2e77)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 3938654f88ce010042e48909e23991e178e411e2bd9e3c9ec25fcb8157c0cd55)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum b026e4d8c11e9ff6f22a7b9e10b8bb29e7572665f0d7978a3cef7d2354b7693f)

Windows Server 2019

Baixe o software versão 5.14.0 para Windows Server 2019 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum ee5a8e8e85fff7acd0bbafa23740e3981f7dc52e708972b600c2b26603786838)
- [Provedor JCE](#) (SHA256 checksum 2ae0274f09f66981c03fd1e3c264e896ba7cd211168ea31369335db1b3ea2e77)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 3938654f88ce010042e48909e23991e178e411e2bd9e3c9ec25fcb8157c0cd55)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum b026e4d8c11e9ff6f22a7b9e10b8bb29e7572665f0d7978a3cef7d2354b7693f)

Windows Server 2016

Baixe o software versão 5.14.0 para Windows Server 2016 na arquitetura x86_64:

- [Biblioteca PKCS #11](#) (SHA256 checksum ee5a8e8e85fff7acd0bbafa23740e3981f7dc52e708972b600c2b26603786838)
- [Provedor JCE](#) (SHA256 checksum 2ae0274f09f66981c03fd1e3c264e896ba7cd211168ea31369335db1b3ea2e77)
 - [Javadocs para AWS CloudHSM](#) (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- [CloudHSM CLI](#) (SHA256 checksum 3938654f88ce010042e48909e23991e178e411e2bd9e3c9ec25fcb8157c0cd55)
- [Provedor de armazenamento de chaves \(KSP\)](#) (SHA256 checksum b026e4d8c11e9ff6f22a7b9e10b8bb29e7572665f0d7978a3cef7d2354b7693f)

O Client SDK 5.14 adiciona suporte para o uso de chaves com controle de quórum e operações de gerenciamento de chaves usando a CloudHSM CLI. O Client SDK 5.14 também adiciona suporte para [Provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#) para plataformas Windows. O Client SDK 5.14 também adiciona pacotes de instalação para a biblioteca PKCS #11, o provedor JCE, a CloudHSM CLI e o Provedor de armazenamento de chaves (KSP) para Windows Server 2022.

Suporte à plataforma

- Inclusão de suporte ao Windows Server 2022 para a biblioteca PKCS #11, provedor JCE, CloudHSM CLI e provedor de armazenamento de chaves (KSP).

CloudHSM CLI

- Inclusão de suporte para o uso de chaves com controle de quórum e operações de gerenciamento de chaves.

Provedor de armazenamento de chaves (KSP)

- Inclusão de suporte para o Provedor de armazenamento de chaves (KSP), uma API de criptografia específica para o sistema operacional Microsoft Windows. Para obter mais informações, consulte [Provedor de armazenamento de chaves \(KSP\) para o AWS CloudHSM Client SDK 5](#).

Versão 5.13.0

O Client SDK 5.13 adiciona suporte à configuração de TLS mútuo nos tipos de cluster hsm2m.medium. Para obter informações sobre como usar o TLS mútuo com o CloudHSM, consulte [Configurar o TLS mútuo entre o cliente e AWS CloudHSM \(recomendado\)](#). O Client SDK 5.13 também adiciona pacotes de instalação para o Ubuntu 24.04 LTS.

Suporte à plataforma

- Foi adicionado suporte para o Ubuntu 24.04 LTS em x86_64 e arquiteturas para todos. ARM64 SDKs

CloudHSM CLI

- Suporte adicionado para usuários administradores executarem o comando [Replicar uma chave com a CLI do CloudHSM](#). O Client SDK 5.12 introduziu o comando key replicate para usuários de criptomoedas.
- O seguinte comando foi incluído:
 - [A categoria cluster mtls na CLI do CloudHSM](#)

Correções de bugs e melhorias:

- Correção de um problema para diminuir o tempo necessário para o cliente detectar conexões HSM não íntegras, o que ajuda a evitar erros de queda de conexão durante a inicialização a quente de lambda.

Versão 5.12.0

O Client SDK 5.12.0 adiciona suporte ARM a várias plataformas e melhorias de desempenho para todos. SDKs Novos atributos foram adicionados ao provedor de CLI e JCE do CloudHSM.

Suporte à plataforma

- Foi adicionado suporte para o Amazon Linux 2023 na ARM64 arquitetura para todos SDKs.
- Foi adicionado suporte para o Red Hat Enterprise Linux 9 (9.2+) na ARM64 arquitetura para todos SDKs
- Foi adicionado suporte para o Ubuntu 22.04 LTS na ARM64 arquitetura para todos SDKs

CloudHSM CLI

- O seguinte comando foi incluído:
 - [Replicar uma chave com a CLI do CloudHSM](#)
- Suporte adicionado para conexão com vários clusters. Para obter mais informações, consulte [Conectar-se a vários clusters com a CLI do CloudHSM](#).

Provedor JCE

- KeyReferenceSpec adicionado para recuperar chaves usando KeyStoreWithAttributes.
- getKeys adicionado para recuperar várias chaves ao mesmo tempo usando KeyStoreWithAttributes.

Melhorias de performance

- Melhorias no desempenho da NoPadding operação AES CBC para todos SDKs.

Versão 5.11.0

O Client SDK 5.11.0 adiciona novos recursos, melhora a estabilidade e inclui correções de erros para todos SDKs

Suporte à plataforma

- Foi adicionado suporte para Amazon Linux 2023 e RHEL 9 (9.2+) para todos SDKs
- Suporte para o Ubuntu 18.04 LTS removido devido ao seu recente fim de vida útil.
- Suporte para Amazon Linux removido devido ao seu recente fim de vida útil.

CloudHSM CLI

- Os seguintes comandos foram adicionados:
 - [A categoria de assinatura criptográfica na CLI do CloudHSM](#)
 - [A categoria de verificação de criptografia na CLI do CloudHSM](#)
 - [Importar uma chave no formato PEM com a CLI do CloudHSM](#)
 - [O comando key unwrap na CLI do CloudHSM](#)
 - [O comando key wrap na CLI do CloudHSM](#)
- Agora o [Exportar uma chave assimétrica com a CloudHSM CLI](#) é compatível com a exportação de chaves públicas.

Mecanismo dinâmico do OpenSSL

- O AWS CloudHSM OpenSSL Dynamic Engine agora é suportado em plataformas que vêm instaladas com uma versão 3.x da biblioteca OpenSSL. Isso inclui Amazon Linux 2023, RHEL 9 (9.2+) e Ubuntu 22.04.

JCE

- Suporte adicionado para JDK 17 e do JDK 21.
- Foi adicionado suporte para chaves AES a serem usadas em operações HMAC.
- Foi adicionado o novo atributo de chave ID.
- Uma nova variante de `DataExceptionCause` foi introduzida para esgotamento de chaves: `DataExceptionCause.KEY_EXHAUSTED`.

Correções de bugs e melhorias:

- O tamanho máximo do atributo `label` foi aumentado de 126 para 127 caracteres.
- Corrigido um bug que impedia o desempacotamento das chaves EC com o RsaOaep mecanismo.
- Foi resolvido um problema conhecido da operação `getKey` no provedor JCE. Para mais detalhes, consulte [Problema: vazamento de memória do Client SDK 5 com operações getKey](#).
- Login aprimorado em todas SDKs as chaves Triple DES que atingiram seu limite máximo de bloqueio de criptografia, de acordo com o FIPS 140-2.

- Foram adicionados os problemas conhecidos do OpenSSL Dynamic Engine. Para mais detalhes, consulte [Problemas conhecidos do OpenSSL Dynamic Engine para AWS CloudHSM](#).

Versão 5.10.0

O Client SDK 5.10.0 melhora a estabilidade e inclui correções de erros para todos. SDKs

CloudHSM CLI

- Adicionados novos comandos que permitem aos clientes gerenciar chaves usando a CloudHSM CLI, incluindo:
 - Criar chaves simétricas e pares de chaves assimétricas
 - Compartilhar e descompartilhar chaves
 - Listar e filtrar chaves usando atributos-chave
 - Listar atributos de chaves
 - Gerar arquivos de referência de chaves
 - Excluir chaves
- Melhoria do registro em log de erros.
- Adicionado suporte para comandos unicode de várias linhas no modo interativo.

Correções de bugs e melhorias:

- Desempenho aprimorado para importar, desempacotar, derivar e criar chaves de sessão para todos. SDKs
- Corrigido um bug no provedor JCE que impedia que arquivos temporários fossem removidos na saída.
- Corrigido um bug que causava um erro de conexão sob certas condições após HSMs a substituição do cluster.
- Formato de saída `getVersion` do JCE modificado para lidar com grandes números de versão menores e incluir o número do patch.

Suporte à plataforma

- Adicionado suporte para o Ubuntu 22.04 com JCE, PKCS #11 e CloudHSM CLI (o suporte para OpenSSL Dynamic Engine ainda não está disponível).

Versão 5.9.0

O Client SDK 5.9.0 melhora a estabilidade e inclui correções de erros para todos. SDKs Uma otimização foi feita para todos, a fim de SDKs informar os aplicativos sobre falhas operacionais imediatamente quando um HSM for determinado como indisponível. Esta versão inclui aprimoramentos de desempenho para JCE.

Provedor JCE

- Desempenho aprimorado
- Corrigido um [problema conhecido](#) de esgotamento do pool de sessões

AWS CloudHSM versões obsoletas do Client SDK

As versões 5.8.0 e anteriores foram descontinuadas. Não recomendamos o uso de versões descontinuadas em workloads de produção. Não fornecemos atualizações compatíveis com versões anteriores para versões descontinuadas, nem hospedamos versões descontinuadas para download. Se você sofrer impacto na produção ao usar versões descontinuadas, deverá atualizar para obter correções de software.

Versões descontinuadas do Client SDK 5

Esta seção lista as versões descontinuadas do Client SDK 5.

Versão 5.8.0

A versão 5.8.0 introduz a autenticação de quórum para a CLI do CloudHSM, suporte ao usuário para JCESSL/TLS offload with JSSE, multi-slot support for PKCS #11, multi-cluster/multi, extração de chaves com JCE, KeyFactory compatível com JCE, novas configurações de repetição para códigos de retorno que não sejam terminais e inclui estabilidade aprimorada e correções de erros para todos. SDKs

Biblioteca PKCS #11

- Adicionado suporte para a configuração de vários slots.

Provedor JCE

- Adicionada a extração de chaves com base na configuração.

- Adicionado suporte para configurações de vários clusters e vários usuários.
- Adicionado suporte para descarregamento de SSL e TLS com JSSE.
- Foi adicionado suporte para desembrulhar para AES/CBC/NoPadding.
- Foram adicionados novos tipos de fábricas importantes: SecretKeyFactory KeyFactory e.

CloudHSM CLI

- Adicionado suporte para autenticação de quórum

Versão 5.7.0

A versão 5.7.0 introduz a CloudHSM CLI e inclui um novo algoritmo de cipher-based message authentication code (CMAC – código de autenticação de mensagens baseado em cifras). Esta versão adiciona a arquitetura ARM no Amazon Linux 2. Os Javadocs do provedor JCE agora estão disponíveis para AWS CloudHSM.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.
- Agora compatível com a arquitetura ARM com o Amazon Linux 2.
- Algoritmos
 - CKM_AES_CMAC (assinar e verificar)

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.
- Agora compatível com a arquitetura ARM com o Amazon Linux 2.

Provedor JCE

- Maior estabilidade e correção de erros.
- Algoritmos
 - ASES_CMAC

Versão 5.6.0

A versão 5.6.0 inclui suporte a novos mecanismos para a biblioteca PKCS #11 e o provedor JCE. Além disso, a versão 5.6 suporta o Ubuntu 20.04.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.
- Mecanismos
 - CKM_RSA_X_509, para modos de criptografia, descriptografia, assinatura e verificação

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Maior estabilidade e correção de erros.
- Cifras
 - RSA/ECB/NoPadding, para modos de criptografia e descriptografia

Chaves compatíveis

- EC com curvas secp224r1 e secp521r1

Suporte à plataforma

- Adicionar suporte para Ubuntu 20.04.

Versão 5.5.0

A versão 5.5.0 adiciona suporte para OpenJDK 11, integração com Keytool e Jarsigner e mecanismos adicionais ao provedor JCE. Resolve um [problema conhecido](#) relacionado a uma KeyGenerator classe que interpreta incorretamente o parâmetro de tamanho da chave como número de bytes em vez de bits.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Suporte para os utilitários Keytool e Jarsigner
- Suporte para OpenJDK 11 em todas as plataformas
- Cifras
 - AES/CBC/NoPaddingModo de criptografia e descriptografia
 - AES/ECB/PKCS5Padding Modo de criptografia e descriptografia
 - AES/CTR/NoPaddingModo de criptografia e descriptografia
 - AES/GCM/NoPaddingModo Embrulhar e Desembrulhar
 - DESede/ECB/PKCS5Padding Modo de criptografia e descriptografia
 - DESede/CBC/NoPaddingModo de criptografia e descriptografia
 - AESWrap/ECB/NoPaddingModo Embrulhar e Desembrulhar
 - AESWrap/ECB/PKCS5Padding Modo Embrulhar e Desembrulhar
 - AESWrap/ECB/ZeroPaddingModo Embrulhar e Desembrulhar
 - RSA/ECB/PKCS1Padding Modo Embrulhar e Desembrulhar
 - RSA/ECB/OAEPPaddingModo Embrulhar e Desembrulhar
 - RSA/ECB/OAEPWithSHA-1 Modo de ANDMGF1 acolchoamento, embalagem e desembalagem
 - RSA/ECB/OAEPWithSHA-224 Modo ANDMGF1 acolchoado, embrulhar e desembrulhar
 - RSA/ECB/OAEPWithSHA-256 Modo ANDMGF1 acolchoado, embrulhar e desembrulhar
 - RSA/ECB/OAEPWithSHA-384 Modo ANDMGF1 acolchoado, embrulhar e desembrulhar
 - RSA/ECB/OAEPWithSHA-512 Modo ANDMGF1 acolchoado, embrulhar e desembrulhar
 - RSAAESWrap/ECB/OAEPPaddingModo Embrulhar e Desembrulhar
 - RSAAESWrap/ECB/OAEPWithSHA-1 Modo de ANDMGF1 acolchoamento, embalagem e desembalagem

- RSAAESWrap/ECB/OAEPWithSHA-224 Modo ANDMGF1 acolchado, embrulhar e desembrulhar
- RSAAESWrap/ECB/OAEPWithSHA-256 Modo ANDMGF1 acolchado, embrulhar e desembrulhar
- RSAAESWrap/ECB/OAEPWithSHA-384 Modo ANDMGF1 acolchado, embrulhar e desembrulhar
- RSAAESWrap/ECB/OAEPWithSHA-512 Modo ANDMGF1 acolchado, embrulhar e desembrulhar
- KeyFactory and SecretKeyFactory
 - RSA chaves RSA de 2048 bits a 4096 bits, em incrementos de 256 bits
 - AES – chaves AES de 128, 192 e 256 bits
 - Pares de chaves do EC para curvas NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1
 - DESede (3 DES)
 - GenericSecret
 - HMAC — com SHA1,,, SHA224 SHA256 SHA384, suporte a SHA512 hash
- Assinar/verificar
 - RSASSA-PSS
 - SHA1withRSA/PSS
 - SHA224withRSA/PSS
 - SHA256withRSA/PSS
 - SHA384withRSA/PSS
 - SHA512withRSA/PSS
 - SHA1withRSAandMGF1
 - SHA224withRSAandMGF1
 - SHA256withRSAandMGF1
 - SHA384withRSAandMGF1
 - SHA512withRSAandMGF1

Versão 5.4.2

A versão 5.4.2 inclui estabilidade aprimorada e correções de erros para todos SDKs. Essa também é a versão mais recente da plataforma CentOS 8. Para obter mais informações, consulte o [site do CentOS](#).

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Maior estabilidade e correção de erros.

Versão 5.4.1

A versão 5.4.1 resolve um [problema conhecido](#) com a biblioteca PKCS #11. Essa também é a versão mais recente da plataforma CentOS 8. Para obter mais informações, consulte o [site do CentOS](#).

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Maior estabilidade e correção de erros.

Versão 5.4.0

A versão 5.4.0 adiciona suporte inicial para o provedor JCE para todas as plataformas. O provedor JCE é compatível com o OpenJDK 8.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Tipos de chave
 - RSA: 2.048 bits para chaves RSA de 4.096 bits, em incrementos de 256 bits.
 - AES: chaves AES de 128, 192 e 256 bits.
 - Pares de chaves do ECC para curvas NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1.
 - DESede (3 DES)
 - HMAC — com suporte SHA1 a SHA512 hash SHA224 SHA256, SHA384,,.
- Cifras (somente criptografar e descriptografar)
 - AES/GCM/NoPadding
 - AES/ECB/NoPadding
 - AES/CBC/PKCS5) Acolchoamento
 - DESede/ECB/NoPadding
 - DESede/CBC/PKCS5) Acolchoamento
 - AES/CTR/NoPadding
 - RSA/ECB/PKCS1 acolchoamento
 - RSA/ECB/OAEPPadding
 - RSA/ECB/OAEPWithSHA-1 ANDMGF1 Acolchoamento
 - RSA/ECB/OAEPWithSHA-224 Acolchoamento ANDMGF1
 - RSA/ECB/OAEPWithSHA-256 Acolchoamento ANDMGF1
 - RSA/ECB/OAEPWithSHA-384 Acolchoamento ANDMGF1
 - RSA/ECB/OAEPWithSHA-512 ANDMGF1 Acolchoamento
- Resumo

- SHA-224
- SHA-256
- SHA-384
- SHA-512
- Assinar/verificar
 - NONEwithRSA
 - SHA1withRSA
 - SHA224withRSA
 - SHA256withRSA
 - SHA384withRSA
 - SHA512withRSA
 - NONEwithECDSA
 - SHA1withECDSA
 - SHA224withECDSA
 - SHA256withECDSA
 - SHA384withECDSA
 - SHA512withECDSA
- Integração com o Java KeyStore

Versão 5.3.0

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Adicione suporte para ECDSA sign/verify com curvas P-256, P-384 e secp256k1.
- Adicione suporte para as plataformas: Amazon Linux, Amazon Linux 2, CentOS 7.8+, RHEL 7 (7.8+).
- Adicione suporte para OpenSSL versão 1.0.2.
- Maior estabilidade e correção de erros.

Provedor JCE

- Tipos de chave
 - RSA: 2.048 bits para chaves RSA de 4.096 bits, em incrementos de 256 bits.
 - AES: chaves AES de 128, 192 e 256 bits.
 - Pares de chaves do EC para curvas NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1 (Blockchain).
 - DESede (3 DES)
 - HMAC — com suporte SHA1 a SHA512 hash SHA224 SHA256, SHA384,,.
- Cifras (somente criptografar e descriptografar)
 - AES/GCM/NoPadding
 - AES/ECB/NoPadding
 - AES/CBC/PKCS5) Acolchoamento
 - DESede/ECB/NoPadding
 - DESede/CBC/PKCS5) Acolchoamento
 - AES/CTR/NoPadding
 - RSA/ECB/PKCS1 acolchoamento
 - RSA/ECB/OAEPPadding
 - RSA/ECB/OAEPWithSHA-1 ANDMGF1 Acolchoamento
 - RSA/ECB/OAEPWithSHA-224 Acolchoamento ANDMGF1
 - RSA/ECB/OAEPWithSHA-256 Acolchoamento ANDMGF1
 - RSA/ECB/OAEPWithSHA-384 Acolchoamento ANDMGF1
 - RSA/ECB/OAEPWithSHA-512 ANDMGF1 Acolchoamento
- Resumo
 - SHA-1
 - SHA-224
 - SHA-256
 - SHA-384
 - SHA-512
- Assinar/verificar
 - NONEwithRSA

- SHA1withRSA
- SHA224withRSA
- SHA256withRSA
- SHA384withRSA
- SHA512withRSA
- NONEwithECDSA
- SHA1withECDSA
- SHA224withECDSA
- SHA256withECDSA
- SHA384withECDSA
- SHA512withECDSA
- Integração com o Java KeyStore

Versão 5.2.1

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Versão 5.2.0

A versão 5.2.0 adiciona suporte a tipos e mecanismos de chaves adicionais à biblioteca PKCS #11.

Biblioteca PKCS #11

Tipos de chave

- ECDSA – curvas P-224, P-256, P-384, P-521 e secp256k1
- Triple DES (3DES)

Mecanismos

- CKM_EC_KEY_PAIR_GEN
- COM_ _KEY_GEN DES3
- COM_ _CBC DES3
- COM_ _CBC_PAD DES3
- COM_ _BCE DES3
- CKM_ECDSA
- CKM_ECDSA_ SHA1
- CKM_ECDSA_ SHA224
- CKM_ECDSA_ SHA256
- CKM_ECDSA_ SHA384
- CKM_ECDSA_ SHA512
- CKM_RSA_PKCS para criptografar/descriptografar

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Versão 5.1.0

A versão 5.1.0 adiciona suporte mecanismos adicionais à biblioteca PKCS #11.

Biblioteca PKCS #11

Mecanismos

- CKM_RSA_PKCS para agrupamento/desagrupamento
- CKM_RSA_PKCS_PSS
- COM_ _RSA_PKCS_PSS SHA1
- COM_ _RSA_PKCS_PSS SHA224
- COM_ _RSA_PKCS_PSS SHA256
- COM_ _RSA_PKCS_PSS SHA384
- COM_ _RSA_PKCS_PSS SHA512
- CKM_AES_ECB

- CKM_AES_CTR
- CKM_AES_CBC
- CKM_AES_CBC_PAD
- COM_ _108_CONTADOR_PDF SP800
- CKM_GENERIC_SECRET_KEY_GEN
- CKM_SHA_1_HMAC
- COM_ _HMAC SHA224
- COM_ _HMAC SHA256
- COM_ _HMAC SHA384
- COM_ _HMAC SHA512
- Somente Wrap/Unwrap CKM_RSA_PKCS_OAEP
- CKM_RSA_AES_KEY_WRAP
- CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD
- COM_CLOUDHSM_AES_KEY_WRAP_ _PAD PKCS5
- CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD

Operações de API

- C_ CreateObject
- C_ DeriveKey
- C_ WrapKey
- C_ UnWrapKey

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Versão 5.0.1

A versão 5.0.1 adiciona suporte inicial ao OpenSSL Dynamic Engine.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Versão inicial do OpenSSL Dynamic Engine.
- Esta versão oferece suporte introdutório para tipos de chaves e OpenSSL APIs:
 - Geração de chave de RSA para chaves de 2048, 3072 e 4096 bits.
 - OpenSSL APIs:
 - [RSA Sign usando RSA](#) PKCS com SHA1 /224/256/384/512 e RSA PSS
 - [Geração de chaves RSA](#)

Para obter mais informações, consulte [OpenSSL Dynamic Engine](#).

- Plataformas suportadas: CentOS 8.3+, Red Hat Enterprise Linux (RHEL) 8.3+ e Ubuntu 18.04 LTS
- Requer: OpenSSL 1.1.1

Para obter mais informações, consulte [Plataformas compatíveis](#).

- Support para SSL/TLS Offload no CentOS 8.3+, Red Hat Enterprise Linux (RHEL) 8.3 e Ubuntu 18.04 LTS, incluindo NGINX 1.19 (para pacotes de criptografia selecionados).

Para obter mais informações, consulte [SSL/TLS Offload on Linux using Tomcat](#) ou [SSL/TLS Offload on Linux using NGINX or Apache](#).

Versão 5.0.0

A versão 5.0.0 é a primeira versão.

Biblioteca PKCS #11

- Esta é a versão inicial.

Suporte introdutório à biblioteca PKCS #11 no Client SDK versão 5.0.0

Esta seção detalha o suporte para tipos de chaves, mecanismos, operações de API e atributos do Client SDK versão 5.0.0.

Tipos de chave:

- AES– chaves AES de 128, 192 e 256 bits
- RSA– chaves RSA de 2048 bits a 4096 bits, em incrementos de 256 bits

Mecanismos:

- CKM_AES_GCM
- CKM_AES_KEY_GEN
- CKM_CLOUDHSM_AES_GCM
- CKM_RSA_PKCS
- CKM_RSA_X9_31_KEY_PAIR_GEN
- CKM_SHA1
- COM__RSA_PKCS_SHA1
- CKM_SHA224
- COM__RSA_PKCS_SHA224
- CKM_SHA256
- COM__RSA_PKCS_SHA256
- CKM_SHA384
- COM__RSA_PKCS_SHA384
- CKM_SHA512
- COM__RSA_PKCS_SHA512

Operações de API:

- C_CloseAllSessions
- C_CloseSession
- C_Decrypt
- C_DecryptFinal
- C_DecryptInit
- C_DecryptUpdate
- C_DestroyObject
- C_Digest
- C_DigestFinal
- C_DigestInit
- C_DigestUpdate
- C_Encrypt

- C_EncryptFinal
- C_EncryptInit
- C_EncryptUpdate
- C_Finalize
- C_FindObjects
- C_FindObjectsFinal
- C_FindObjectsInit
- C_GenerateKey
- C_GenerateKeyPair
- C_GenerateRandom
- C_GetAttributeValue
- C_GetFunctionList
- C_GetInfo
- C_GetMechanismInfo
- C_GetMechanismList
- C_GetSessionInfo
- C_GetSlotInfo
- C_GetSlotList
- C_GetTokenInfo
- C_Initialize
- C_Login
- C_Logout
- C_OpenSession
- C_Sign
- C_SignFinal
- C_SignInit
- C_SignUpdate
- C_Verify
- C_VerifyFinal
- C_VerifyInit

- C_VerifyUpdate

Atributos:

- GenerateKeyPair
 - Todos os atributos da chave RSA
- GenerateKey
 - Todos os atributos da chave AES
- GetAttributeValue
 - Todos os atributos da chave RSA
 - Todos os atributos da chave AES

Amostras:

- [Gerar chaves \(AES, RSA, EC\)](#)
- [Listar atributos de chave](#)
- [Criptografar e descriptografar dados com AES-GCM](#)
- [Assinar e verificar dados com RSA](#)

Versões descontinuadas do Client SDK 3

Esta seção lista as versões anteriores do Client SDK 3.

Versão 3.4.4

A versão 3.4.4 adiciona atualizações ao provedor JCE.

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência.

Biblioteca PKCS #11

- Versão atualizada para fins de consistência.

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência.

Provedor JCE

- Atualizar o log4j para a versão 2.17.1.

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência.

Versão 3.4.3

A versão 3.4.3 adiciona atualizações ao provedor JCE.

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência.

Biblioteca PKCS #11

- Versão atualizada para fins de consistência.

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência.

Provedor JCE

- Atualizar o log4j para a versão 2.17.0.

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência.

Versão 3.4.2

A versão 3.4.2 adiciona atualizações ao provedor JCE.

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência.

Biblioteca PKCS #11

- Versão atualizada para fins de consistência.

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência.

Provedor JCE

- Atualizar o log4j para a versão 2.16.0.

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência.

Versão 3.4.1

A versão 3.4.1 adiciona atualizações ao provedor JCE.

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência.

Biblioteca PKCS #11

- Versão atualizada para fins de consistência.

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência.

Provedor JCE

- Atualizar o log4j para a versão 2.15.0.

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência.

Versão 3.4.0

A versão 3.4.0 adiciona atualizações a todos os componentes.

AWS CloudHSM Software cliente

- Maior estabilidade e correção de erros.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Maior estabilidade e correção de erros.

Windows (provedores CNG e KSP)

- Maior estabilidade e correção de erros.

Versão 3.3.2

A versão 3.3.2 resolve um [problema](#) com o script client_info.

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência.

Biblioteca PKCS #11

- Versão atualizada para fins de consistência.

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência.

Provedor JCE

- Versão atualizada para fins de consistência.

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência.

Versão 3.3.1

A versão 3.3.1 adiciona atualizações a todos os componentes.

AWS CloudHSM Software cliente

- Maior estabilidade e correção de erros.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Maior estabilidade e correção de erros.

Windows (provedores CNG e KSP)

- Maior estabilidade e correção de erros.

Versão 3.3.0

A versão 3.3.0 adiciona autenticação de dois fatores (2FA) e outras melhorias.

AWS CloudHSM Software cliente

- Autenticação 2FA adicionada para crypto officers (CO – responsáveis pela criptografia). Para obter mais informações, consulte [Gerenciar a autenticação de dois fatores para responsáveis pela criptografia](#).
- Suporte de plataforma removido para RedHat Enterprise Linux 6 e CentOS 6. Para obter mais informações, consulte [Suporte do Linux](#).
- Adicionada uma versão autônoma do CMU para uso com Client SDK 5 ou Client SDK 3. Essa é a mesma versão do CMU incluída no daemon do cliente da versão 3.3.0 e agora é possível baixar o CMU sem baixar o daemon do cliente.

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.
- Suporte de plataforma removido para RedHat Enterprise Linux 6 e CentOS 6. Para obter mais informações, consulte [Suporte do Linux](#).

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência
- Suporte de plataforma removido para RedHat Enterprise Linux 6 e CentOS 6. Para obter mais informações, consulte [Suporte do Linux](#).

Provedor JCE

- Maior estabilidade e correção de erros.
- Suporte de plataforma removido para RedHat Enterprise Linux 6 e CentOS 6. Para obter mais informações, consulte [Suporte do Linux](#).

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência

Versão 3.2.1

A versão 3.2.1 adiciona uma análise de conformidade entre a AWS CloudHSM implementação da biblioteca PKCS #11 e o padrão PKCS #11, novas plataformas e outras melhorias.

AWS CloudHSM Software cliente

- Adicione suporte de plataforma para CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Para ter mais informações, consulte [???](#).

Biblioteca PKCS #11

- [Relatório de conformidade da biblioteca PKCS #11 para o SDK 3.2.1 do cliente](#)
- Adicione suporte de plataforma para CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Para obter mais informações, consulte [???](#).

Mecanismo dinâmico do OpenSSL

- Não suporta para CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Para obter mais informações, consulte [???](#).

Provedor JCE

- Adicione suporte de plataforma para CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Para obter mais informações, consulte [???](#).

Windows (provedores CNG e KSP)

- Maior estabilidade e correção de erros.

Versão 3.2.0

A versão 3.2.0 adiciona suporte para mascarar senhas e outras melhorias.

AWS CloudHSM Software cliente

- Adiciona suporte para ocultar sua senha ao usar ferramentas de linha de comando. Para obter mais informações, consulte [loginHSM e logoutHSM](#) (cloudhsm_mgmt_util) e [loginHSM e logoutHSM](#) (key_mgmt_util).

Biblioteca PKCS #11

- Adiciona suporte para hash de grandes dados em software para alguns mecanismos PKCS #11 que não eram suportados anteriormente. Para obter mais informações, consulte [Versões compatíveis](#).

Mecanismo dinâmico do OpenSSL

- Maior estabilidade e correção de erros.

Provedor JCE

- Versão atualizada para fins de consistência.

Windows (provedores CNG e KSP)

- Maior estabilidade e correção de erros.

Versão 3.1.2

A versão 3.1.2 adiciona atualizações ao provedor JCE.

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência

Biblioteca PKCS #11

- Versão atualizada para fins de consistência

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência

Provedor JCE

- Atualizar a versão do log4j para 2.13.3

Windows (provedores CNG e KSP)

- Versão atualizada para fins de consistência

Versão 3.1.1

AWS CloudHSM Software cliente

- Versão atualizada para fins de consistência.

Biblioteca PKCS #11

- Versão atualizada para fins de consistência.

Mecanismo dinâmico do OpenSSL

- Versão atualizada para fins de consistência.

Provedor JCE

- Correções de erros e melhorias na performance.

Windows (CNG, KSP)

- Versão atualizada para fins de consistência.

Versão 3.1.0

A versão 3.1.0 adiciona [empacotamento de chaves AES compatível com os padrões](#).

AWS CloudHSM Software cliente

- Um novo requisito para atualização: a versão do cliente deve corresponder à versão de todas as bibliotecas de software que você está usando. Para atualizar, é necessário usar um comando em

lote que atualiza o cliente e todas as bibliotecas ao mesmo tempo. Para obter mais informações, consulte [Atualização do DK3 do cliente](#).

- Key_mgmt_util (KMU) inclui as seguintes atualizações:
 - Adição de dois novos métodos de empacotamento de chaves AES: empacotamento de chaves AES compatível com os padrões com preenchimento de zeros e empacotamento de chave AES sem preenchimento. Para obter mais informações, consulte [wrapKey](#) e [unwrapKey](#).
 - Capacidade desativada de especificar IV personalizado ao encapsular uma chave usando PKCS5 AES_KEY_WRAP_PAD_. Para obter mais informações, consulte [Empacotamento de chaves AES](#).

Biblioteca PKCS #11

- Adição de dois novos métodos de empacotamento de chaves AES: empacotamento de chaves AES compatível com os padrões com preenchimento de zeros e empacotamento de chave AES sem preenchimento. Para obter mais informações, consulte [Empacotamento de chaves AES](#).
- É possível configurar o comprimento de salt para assinaturas RSA-PSS. Para saber como usar esse recurso, consulte [Comprimento de sal configurável para assinaturas RSA-PSS](#) em GitHub

Mecanismo dinâmico do OpenSSL

- ALTERAÇÃO IMPORTANTE: os pacotes de criptografia TLS 1.0 e 1.2 não SHA1 estão disponíveis no OpenSSL Engine 3.1.0. Esse problema será resolvido em breve.
- Se você pretende instalar a biblioteca OpenSSL Dynamic Engine no RHEL 6 ou no CentOS 6, consulte um [problema conhecido](#) sobre a versão padrão do OpenSSL instalada nesses sistemas operacionais.
- Maior estabilidade e correção de erros

Provedor JCE

- MUDANÇA SIGNIFICATIVA: Para resolver um problema de conformidade com a Extensão de Criptografia Java (JCE), o AES wrap and unwrap agora usa corretamente o AESWrap algoritmo em vez do algoritmo AES. Isso significa que Cipher.WRAP_MODE os Cipher.UNWRAP_MODE mecanismos AES/CBC AES/ECB não são mais bem-sucedidos.

Para atualizar para a versão de cliente 3.1.0, é necessário atualizar o código. Se você tiver chaves encapsuladas existentes, será necessário prestar bastante atenção ao mecanismo usado para

desencapsular e como os padrões IV foram alterados. Se você empacotou as chaves com a versão 3.0.0 ou anterior do cliente, na versão 3.1.1 você deve usar AESWrap/ECB/PKCS5Padding para desempacotar as chaves existentes. Para obter mais informações, consulte [Empacotamento de chaves AES](#).

- É possível listar diversas chaves com o mesmo rótulo do provedor JCE. Para saber como iterar em todas as chaves disponíveis, consulte [Localizar todas as chaves](#) em GitHub.
- É possível definir valores mais restritivos para atributos durante a criação da chave, incluindo especificar rótulos diferentes para chaves públicas e privadas. Para obter mais informações, consulte [Atributos Java compatíveis](#).

Windows (CNG, KSP)

- Maior estabilidade e correção de erros.

AWS CloudHSM end-of-life Lançamentos do SDK do cliente

As seguintes versões do AWS CloudHSM cliente chegaram ao fim do suporte. Essas versões do cliente do AWS CloudHSM não são mais compatíveis com o serviço e não receberão atualizações. Para preservar a segurança do seu aplicativo, AWS CloudHSM pode recusar conexões de versões que chegaram ao fim do suporte.

- As versões 3.4.4 e anteriores do SDK chegaram ao fim do suporte.
- As versões 5.8.0 e anteriores do SDK chegaram ao fim do suporte.

Histórico do documento

Este tópico descreve atualizações importantes no Guia do usuário do AWS CloudHSM .

Tópicos

- [Atualizações recentes](#)
- [Atualizações anteriores](#)

Atualizações recentes

A tabela a seguir descreve alterações significativas nesta documentação desde abril de 2018. Além das principais alterações listadas aqui, também atualizamos a documentação com frequência para melhorar as descrições e os exemplos e abordar os comentários que você nos envia. Para ser notificado sobre alterações significativas, use o link no canto superior direito para assinar os feeds RSS.

Para obter detalhes sobre novos lançamentos, consulte [Downloads para o SDK AWS CloudHSM do cliente](#)

Alteração	Descrição	Data
Adicionado lançamento	Lançou a versão 5.17.0 do AWS CloudHSM cliente.	6 de janeiro de 2026
Links de download do OpenSSL Provider adicionados	Foram adicionados links de download do OpenSSL Provider AWS CloudHSM para a versão 5.17 do cliente.	6 de janeiro de 2026
Adicionado lançamento	Lançou a versão 5.16.2 do AWS CloudHSM cliente.	16 de setembro de 2025
Adicionado lançamento	Lançou a versão 5.16.1 do AWS CloudHSM cliente.	25 de junho de 2025
Adicionado lançamento	Lançou a versão 5.16.0 do AWS CloudHSM cliente.	1.º de maio de 2025

Adicionado lançamento	Lançou a versão 5.15.0 do AWS CloudHSM cliente.	3 de fevereiro de 2025
Adicionado lançamento	Lançou a versão 5.14.0 do AWS CloudHSM cliente.	26 de novembro de 2024
Novo modo de cluster e tipo de HSM	Foi adicionado o suporte à criação (hsm2m.medium) em clusters no modo FIPS.	20 de agosto de 2024
Adicionado lançamento	Lançou a versão 5.13.0 do AWS CloudHSM cliente.	13 de agosto de 2024
Novo modo de cluster e tipo de HSM	Lançado um novo tipo de HSM (hsm2m.medium) e um novo modo de cluster (não FIPS).	10 de junho de 2024
Adicionado lançamento	Lançou a versão 5.12.0 do AWS CloudHSM cliente.	20 de março de 2024
Adicionado lançamento	Lançou a versão 5.11.0 do AWS CloudHSM cliente.	17 de janeiro de 2024
Adicionado lançamento	Lançou a versão 5.10.0 do AWS CloudHSM cliente.	28 de julho de 2023
Adicionado lançamento	Lançou a versão 5.9.0 do AWS CloudHSM cliente.	23 de maio de 2023
Adicionado lançamento	Lançou a versão 5.8.0 do AWS CloudHSM cliente.	16 de março de 2023
Adicionado lançamento	Lançou a versão 5.7.0 do AWS CloudHSM cliente.	16 de novembro de 2022
Adicionado lançamento	Lançou a versão 5.6.0 do AWS CloudHSM cliente.	1º de setembro de 2022
Adicionado lançamento	Lançou a versão 5.5.0 do AWS CloudHSM cliente.	13 de maio de 2022

Adicionado lançamento	Lançou a versão 5.4.2 do AWS CloudHSM cliente.	18 de março de 2022
Adicionado lançamento	Lançou a versão 5.4.1 do AWS CloudHSM cliente.	10 de fevereiro de 2022
Adicionado lançamento	Lançou a versão 5.4.0 do provedor AWS CloudHSM JCE para plataformas Windows.	1º de fevereiro de 2022
Adicionado lançamento	Lançou a versão 5.4.0 do AWS CloudHSM cliente, que adiciona suporte inicial ao provedor JCE para todas as plataformas Linux.	28 de janeiro de 2022
Adicionado lançamento	Lançou a versão 5.3.0 do AWS CloudHSM cliente.	3 de janeiro de 2022
Adicionado lançamento	Lançou a versão 3.4.4 do AWS CloudHSM cliente.	3 de janeiro de 2022
Adicionado lançamento	Lançou a versão 3.4.3 do AWS CloudHSM cliente.	20 de dezembro de 2021
Adicionado lançamento	Lançou a versão 3.4.2 do AWS CloudHSM cliente.	15 de dezembro de 2021
Adicionado lançamento	Lançou a versão 3.4.1 do AWS CloudHSM cliente.	10 de dezembro de 2021
Adicionado lançamento	Lançou a versão 5.2.1 do AWS CloudHSM cliente.	4 de outubro de 2021
Adicionado lançamento	Lançou a versão 3.4.0 do AWS CloudHSM cliente.	25 de agosto de 2021

Adicionado lançamento	Lançou a versão 5.2.0 do AWS CloudHSM cliente.	3 de agosto de 2021
Adicionado lançamento	Lançou a versão 3.3.2 do AWS CloudHSM cliente.	2 de julho de 2021
Adicionado lançamento	Lançou a versão 5.1.0 do AWS CloudHSM cliente.	1º de junho de 2021
Adicionado lançamento	Lançou a versão 3.3.1 do AWS CloudHSM cliente.	26 de abril de 2021
Adicionado lançamento	Lançou a versão 5.0.1 do AWS CloudHSM cliente.	8 de abril de 2021
Adicionado lançamento	Lançou a versão 5.0.0 do AWS CloudHSM cliente.	12 de março de 2021
Novo conteúdo adicionado	Interface adicionada VPC endpoint, um recurso da AWS que permite criar uma conexão privada entre sua VPC AWS CloudHSM sem exigir acesso pela Internet ou por meio de um dispositivo NAT, uma conexão VPN ou uma conexão. Direct Connect	10 de fevereiro de 2021
Adicionado lançamento	Lançou a versão 3.3.0 do AWS CloudHSM cliente.	3 de fevereiro de 2021
Adicionar novo conteúdo	Adicionada retenção gerenciada de backup, um atributo que exclui backups antigos automaticamente.	18 de novembro de 2020

Adicionar novo conteúdo	Foi adicionado um relatório de conformidade que analisa a implementação do AWS CloudHSM Client SDK 3.2.1 da biblioteca PKCS #11 com o padrão PKCS #11.	29 de outubro de 2020
Adicionado lançamento	Lançou a versão 3.2.1 do AWS CloudHSM cliente.	8 de outubro de 2020
Novo conteúdo adicionado	Adicionada documentação que descreve as principais configurações de sincronização no AWS CloudHSM.	1.º de setembro de 2020
Adicionado lançamento	Lançou a versão 3.2.0 do AWS CloudHSM cliente.	31 de agosto de 2020
Adicionado lançamento	Lançou a versão 3.1.2 do AWS CloudHSM cliente.	30 de julho de 2020
Adicionado lançamento	Lançou a versão 3.1.1 do AWS CloudHSM cliente.	3 de junho de 2020
Adicionado lançamento	Lançou a versão 3.1.0 do AWS CloudHSM cliente.	21 de maio de 2020
Adicionado lançamento	Lançou a versão 3.0.1 do AWS CloudHSM cliente.	20 de abril de 2020
Adicionado lançamento	Lançou a versão 3.0.0 do AWS CloudHSM cliente para a plataforma Windows Server.	30 de outubro de 2019
Adicionado lançamento	Lançou a versão 3.0.0 do AWS CloudHSM cliente para todas as plataformas, exceto Windows.	22 de outubro de 2019

Adicionado lançamento	Lançou a versão 2.0.4 do AWS CloudHSM cliente.	26 de agosto de 2019
Adicionado lançamento	Lançou a versão 2.0.3 do AWS CloudHSM cliente.	13 de maio de 2019
Adicionado lançamento	Lançou a versão 2.0.1 do AWS CloudHSM cliente.	21 de março de 2019
Adicionado lançamento	Lançou a versão 2.0.0 do AWS CloudHSM cliente.	6 de fevereiro de 2019
Adição de suporte à região	AWS CloudHSM Suporte adicional para as regiões da UE (Estocolmo) e AWS GovCloud (Leste dos EUA).	19 de dezembro de 2018
Adicionado lançamento	Lançou a versão 1.1.2 do AWS CloudHSM cliente para Windows.	20 de novembro de 2018
Problemas conhecidos atualizados	Foi adicionado novo conteúdo ao guia Solução de problemas .	8 de novembro de 2018
Adicionado lançamento	Lançou a versão 1.1.2 do AWS CloudHSM cliente para plataformas Linux.	8 de novembro de 2018
Adição de suporte à região	Foi adicionado AWS CloudHSM suporte para as regiões da UE (Paris) e Ásia-Pacífico (Seul).	24 de outubro de 2018
Novo conteúdo adicionado	Foi adicionada a capacidade de excluir e restaurar AWS CloudHSM backups.	10 de setembro de 2018

Novo conteúdo adicionado	Foi adicionada a entrega automática de registros de auditoria ao Amazon CloudWatch Logs.	13 de agosto de 2018
Novo conteúdo adicionado	Foi adicionada a capacidade de copiar um backup de AWS CloudHSM cluster entre regiões.	30 de julho de 2018
Adição de suporte à região	Foi adicionado AWS CloudHSM suporte para a região da UE (Londres).	13 de junho de 2018
Novo conteúdo adicionado	Foi adicionado suporte de AWS CloudHSM cliente e biblioteca para Amazon Linux 2, Red Hat Enterprise Linux (RHEL) 6, Red Hat Enterprise Linux (RHEL) 7, CentOS 6, CentOS 7 e Ubuntu 16.04 LTS.	10 de maio de 2018
Adicionado lançamento	Adicionou um AWS CloudHSM cliente Windows.	30 de abril de 2018

Atualizações anteriores

A tabela a seguir descreve as mudanças importantes em relação às AWS CloudHSM anteriores a 2018.

Alteração	Descrição	Data
Novo conteúdo	Autenticação de quórum adicionada (controle de acesso M ou N) para agentes criptográficos (COs). Para	9 de novembro de 2017

Alteração	Descrição	Data
	obter mais informações, consulte Como usar o CloudHSM Management Utility (CMU) para gerenciar a autenticação de quórum (controle de acesso M ou N) .	
Atualizar	Adição de uma documentação sobre como usar a ferramenta de linha de comando key_mgmt_util. Para obter mais informações, consulte Referência para comandos do AWS CloudHSM Key Management Utility .	9 de novembro de 2017
Novo conteúdo	Adição do Oracle Transparent Data Encryption. Para obter mais informações, consulte Criptografia do Oracle Database .	25 de outubro de 2017
Novo conteúdo	Adição de descarregamento SSL. Para obter mais informações, consulte Descarregamento de SSL/TLS .	12 de outubro de 2017
Novo guia	Esta versão apresenta AWS CloudHSM	14 de agosto de 2017

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.