Manual do usuário

AWS CloudHSM

AWS CloudHSM: Manual do usuário

Table of Contents

O que AWS CloudHSMé

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 os HSMs de alta disponibilidade que estão na nuvem da AWS, têm 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 aprovados pelo Federal Information Processing Standard (FIPS) podem ser usados. O modo não FIPS oferece todas as chaves e algoritmos suportados AWS CloudHSM, independentemente da aprovação do FIPS. Para ter mais informações, consulte AWS CloudHSM modos de cluster e tipos de HSM.

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

> AWS CloudHSM usa HSMs de uso geral que oferecem mais flexibilidade quando comparados aos serviços totalmente gerenciados da AWS que têm algoritmos e comprimentos de chave predeterminados para seu aplicativo. Oferecemos HSMs compatíveis com os padrões, de inquilino único e validados pelo FIPS 140-2 de nível 3 para clusters no modo FIPS. Para clientes com casos de uso fora das restrições da validação FIPS 140-2 de nível 3, AWS CloudHSM também oferece clusters no modo não FIPS. Consulte AWS CloudHSM aglomerados Para 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 os SDKs oferecem 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, consulteConceitos básicos.

> **Note**
>
> Se deseja um serviço gerenciado para a criação e controle das chaves de criptografia, mas não quer ou não precisa operar seu próprio HSM, considere usar o AWS Key Management Service.
> Se você estiver procurando por um serviço flexível que gerencie HSMs e chaves de pagamento para aplicativos de processamento de pagamentos na nuvem, considere usar a AWS Payment Cryptography.

Conteúdo

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 pelo 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 ter mais informações, consulte Conformidade.

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 SDKs criptográficos.

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.

Autentique mensagens usando HMACs e CMACs

Na criptografia, os – códigos de autenticação de mensagens cifradas (CMACs) e os códigos de autenticação de mensagens por 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 compatíveis com HMACs e CMACs.

Aproveite os benefícios de AWS CloudHSM e AWS Key Management Service

Os clientes podem combinar AWS CloudHSM e AWS KMSarmazenar materiais essenciais em um ambiente de inquilino único com certificação FIPS 140-2 de nível 3 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 .

Descarregar o processamento SSL/TLS para servidores web

Para enviar dados com segurança pela Internet, os servidores web usam pares de chaves públicas-privadas e certificados de chaves públicas SSL/TLS 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 descarregamento de SSL/TLS com, consulte. AWS CloudHSMDescarregamento 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. É possível obter maior segurança armazenando a chave de criptografia mestra do TDE em HSMs no seu AWS CloudHSM. Para obter informações sobre como configurar o Oracle TDE com AWS CloudHSM, consulteCriptografia 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 usar seus 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 com segurança números aleatórios nos HSMs que você controla 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 crie um cluster, adicione HSMs a ele, crie usuários e chaves e, em seguida, use os SDKs

do cliente para integrar seus 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 do HSM
- Chaves HSM
- Client SDKs
- AWS CloudHSM backups de cluster
- Regiões

AWS CloudHSM aglomerados

Fazer com que HSMs individuais trabalhem juntos 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 (HSMs) em clusters. Um cluster é uma coleção de HSMs individuais que se AWS CloudHSM mantém sincronizados. Ao executar uma tarefa ou operação em um HSM em um cluster, os outros HSMs nesse cluster são atualizados automaticamente.

AWS CloudHSM oferece clusters em dois modos: FIPS e não FIPS. No modo FIPS, somente chaves e algoritmos aprovados pelo Federal Information Processing Standard (FIPS) 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, consulteAWS CloudHSM modos de cluster e tipos de HSM.

Para atender aos seus objetivos 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 colocar os HSMs em diferentes zonas de disponibilidade em uma AWS região. Adicionar mais HSMs em um cluster fornece maior desempenho. Espalhar clusters pelas zonas de disponibilidade fornece redundância e alta disponibilidade.

Para obter mais informações sobre clusters, consulte Gerenciando AWS CloudHSM clusters.

Para criar um cluster, consulte Conceitos básicos.

Usuários do HSM

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 cluster. Em vez disso, você usa usuários do HSM diretamente nos HSMs do 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 HSMs interagindo com recursos 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).

Chaves HSM

O AWS CloudHSM permite que você gere, armazene e gerencie com segurança suas chaves de criptografia em HSMs de locatário único que estão em seu cluster do 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 o 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.
- Usar funções de hash criptográficas para computar resumos de mensagens e códigos de autenticação de mensagem 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 de HSMs que estão no cluster. Por exemplo, hsm2m.medium armazena mais chaves do que hsm1, medium. Para uma comparação, consulteAWS 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 AWS CloudHSM SDKs do cliente.

Como gerenciar chaves

AWS CloudHSM as chaves são gerenciadas por meio de SDKs e ferramentas de linha de comando. Para obter informações sobre como usar essas ferramentas para gerenciar chaves, consulte Gerenciando chaves em AWS CloudHSM e Práticas recomendadas 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 comandos key share e key unshare para compartilhar e cancelar o compartilhamento da chave com outros CUs. Para ter mais informações, consulte Usar a CLI do CloudHSM para compartilhar e cancelar o compartilhamento de chaves.

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.

Client SDKs

Ao usar AWS CloudHSM, você executa operações criptográficas com os kits de desenvolvimento de software (SDKs) do AWS CloudHSM cliente. AWS CloudHSM Os SDKs do cliente incluem:

- Padrão de criptografia de chave pública Nº 11 (PKCS #11)
- Provedor JCE
- Mecanismo dinâmico do OpenSSL
- Cryptography API: Next Generation (CNG – API de criptografia da próxima geração) e provedor de armazenamento de chaves (KSP) para Microsoft Windows

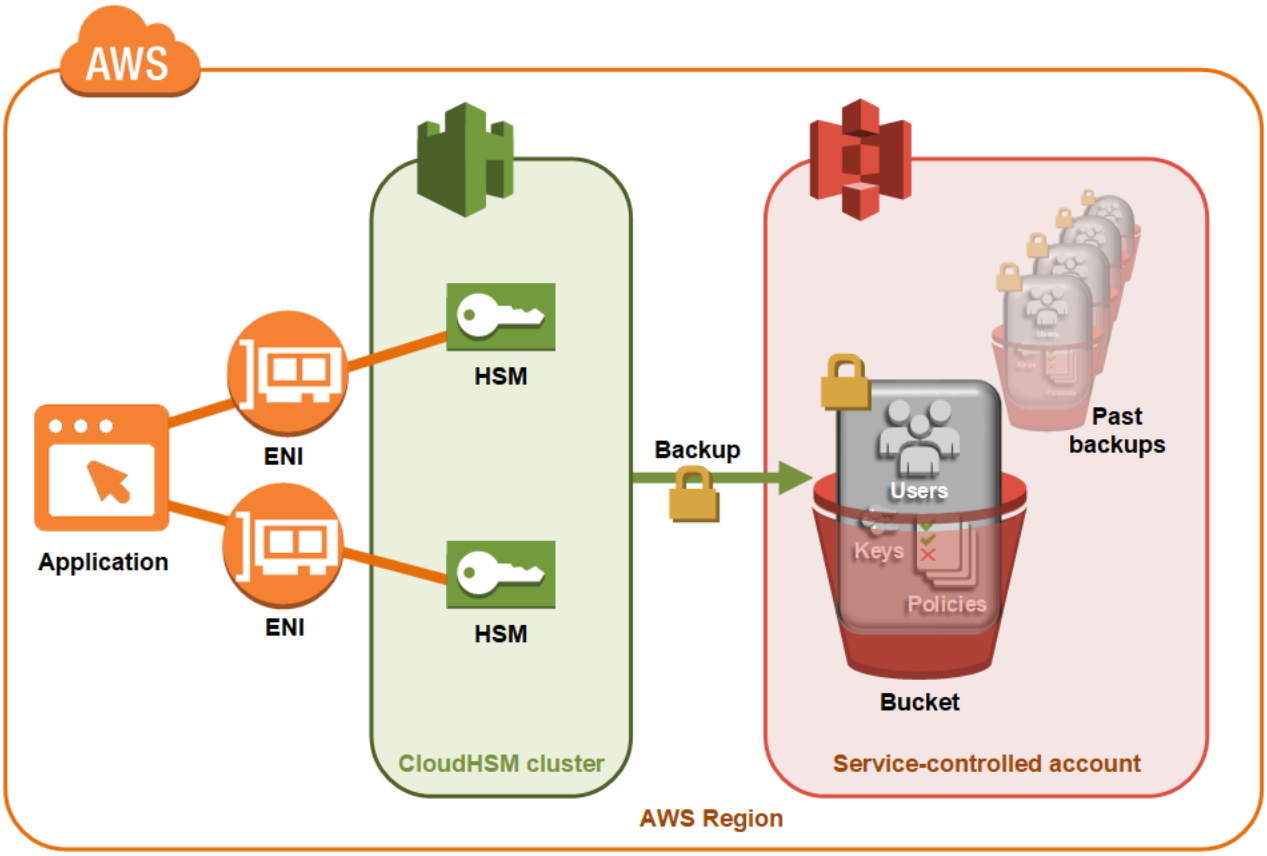
Você pode usar qualquer um ou todos esses SDKS em seu AWS CloudHSM cluster. Escreva o código do seu aplicativo para usar esses SDKs para realizar operações criptográficas nos seus HSMs. Para ver quais plataformas e tipos de HSM são compatíveis com cada SDK, consulte Plataformas compatíveis com o Client SDK 5

Ferramentas utilitárias e de linha de comando são necessárias não apenas para usar SDKs, mas também para definir as credenciais, políticas e configurações do seu aplicativo. Para 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 Client SDKs e. nd-to-end Criptografia E

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 Gerenciar backups.

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 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

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 balanceia AWS CloudHSM automaticamente a carga em todos os HSMs em um cluster.

Como a maioria dos AWS recursos, clusters e HSMs 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 ter mais informações, consulte AWS CloudHSM backups de cluster.

Definição de preço

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 AWS CloudHSM cluster. 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.

Conteúdo

Criar grupos administrativos 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. AWS 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 ter 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.

Inscreva-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 https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

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

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um 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 AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, digite sua senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Signing in as the root user (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. Habilitar o IAM Identity Center.

   Para obter instruções, consulte Habilitar AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .
2. No Centro de Identidade do IAM, conceda o acesso administrativo para 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 seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  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 para 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 Create a permission set no Guia do usuário do AWS IAM Identity Center .

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

   Para obter instruções, consulte Add groups 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, consulteGerenciamento de identidade e acesso para AWS CloudHSM.

Crie uma nuvem privada virtual (VPC).

Se você ainda não tem uma nuvem privada virtual (VPC), siga as etapas neste tópico para criá-la.

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

Para criar uma VPC

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

Criar um cluster

Um cluster é uma coleção de HSMs individuais. 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 em seu cluster. Para obter detalhes sobre as diferenças entre cada tipo de HSM e modo de cluster, consulteAWS CloudHSM modos de cluster e tipos de HSM.

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 aos HSMs no cluster. Ele permite apenas as conexões de entrada das instâncias do Amazon Elastic Compute Cloud (Amazon EC2) 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.

> **Important**
>
> Quando você cria um cluster, AWS CloudHSM cria uma função vinculada ao serviço chamada. AWSServiceRoleForCloudHSM Se você AWS CloudHSM não conseguir criar a função ou se a função ainda não existir, talvez você não consiga criar um cluster. Para ter mais informações, consulte Resolver falhas na criação do cluster. Para obter mais informações sobre funções vinculadas ao serviço, consulte Funções vinculadas a serviços para AWS CloudHSM.

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

> **Note**
>
> Para obter detalhes sobre argumentos de cluster e APIs, consulte create-clusterna AWS CLI Command Reference.

Para criar um cluster (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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 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)..

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 AWSpara 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 é criado, o tipo de HSM e o modo de cluster não podem ser alterados. Para obter informações sobre qual tipo e modo são adequados para seu caso de uso, consulteAWS CloudHSM modos de cluster e tipos de HSM.

d. Em Origem do cluster, especifique se você deseja criar um novo cluster ou restaurar um a partir de um backup existente.

- Os backups de clusters no modo não FIPS só podem ser usados para restaurar clusters que estejam no modo não FIPS.
- Os backups de clusters no modo FIPS só podem ser usados para restaurar clusters que estejam no modo FIPS.

5. Escolha Next (Próximo).
6. Especifique por quanto tempo o serviço deve reter os backups.

> **Note**
>
> 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 ter 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).

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 do backup e os IDs das sub-redes nas quais você planeja criar HSMs. Use os IDs de sub-rede das sub-redes privadas que você criou. Especifique apenas uma sub-rede por zona de disponibilidade.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm1.medium \
      --backup-retention-policy Type=DAYS,Value=<number of days> \
      --subnet-ids <subnet ID>

{
    "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",
        "Certificates": {},
        "State": "CREATE_IN_PROGRESS",
        "Hsms": [],
        "ClusterId": "cluster-igklspoyj5v",
        "ClusterMode": "FIPS",
```

```
        "CreateTimestamp": 1502423370.069
    }
}
```

> **Note**
>
> `ClusterMode`usa como padrão o modo FIPS se não for especificado. Para criar um cluster não FIPS, você deve incluir o parâmetro: `--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
> ```

Para criar um cluster (AWS CloudHSM API)

- Envie uma solicitação CreateCluster. Especifique o tipo de instância do HSM, apolítica de retenção do backup e os IDs de sub-rede das sub-redes nas quais você planeja criar HSMs. Use os IDs de sub-rede das sub-redes privadas que você criou. Especifique apenas uma sub-rede por zona de disponibilidade.

Se as suas tentativas de criar um cluster falharem, isso pode estar relacionado a problemas com as funções vinculadas a serviços do AWS CloudHSM . Para obter ajuda para resolver essa falha, consulte Resolver falhas na criação do cluster.

Rever grupo de segurança do cluster

Quando você cria um cluster, AWS CloudHSM cria um grupo de segurança com o nome`cloudhsm-cluster-`*`clusterID`*`-sg`. 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 com HSMs em seu cluster.

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 a seus HSMs.
- O grupo de segurança do cluster impede o acesso não autorizado aos HSMs. Qualquer pessoa que possa acessar as instâncias no grupo de segurança pode acessar seus HSMs. A maioria das operações exige que um usuário faça login no HSM. No entanto, é possível zerar os HSMs sem autenticação, o que destrói o material de 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.

Na próxima etapa, você poderá lançar uma instância do Amazon EC2 e conectá-la a seus HSMs anexando o grupo de segurança do cluster a ela.

Iniciar uma instância do Amazon EC2.

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
- AWS 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 melhores práticas 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 ter mais informações, consulte 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. Escolha Select an existing security group (Selecionar um grupo de segurança existente).

12. Em Grupos de segurança comuns, selecione o grupo de segurança padrão no menu suspenso.
13. Em Configurar armazenamento, use os menus suspensos para escolher uma configuração de armazenamento.
14. 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.

Configurar os grupos de segurança da instância do cliente Amazon EC2

Quando executou uma instância do Amazon EC2, 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.

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).

Conecte a instância do Amazon EC2 ao cluster AWS CloudHSM

Anexe o grupo de segurança do cluster à instância do EC2 para que a instância do EC2 possa se comunicar com os HSMs em 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.

Criar um HSM

Depois de criar um cluster, você pode criar um 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 DescribeClusterssolicitação na AWS CloudHSM API. Você pode criar um HSM do console do AWS CloudHSM, a AWS CLI ou a API do AWS CloudHSM .

Para criar um HSM (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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.

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",
        "AvailabilityZone": "us-west-2a",
```

```
        "ClusterId": "cluster-igklspoyj5v",
        "EniId": "eni-5d7ade72",
        "SubnetId": "subnet-fd54af9b",
        "State": "CREATE_IN_PROGRESS"
    }
}
```

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.

Verificar a identidade e a autenticidade do HSM de seu cluster (opcional)

Para inicializar seu cluster, assine uma solicitação de assinatura de certificado (CSR) gerada pelo primeiro 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.

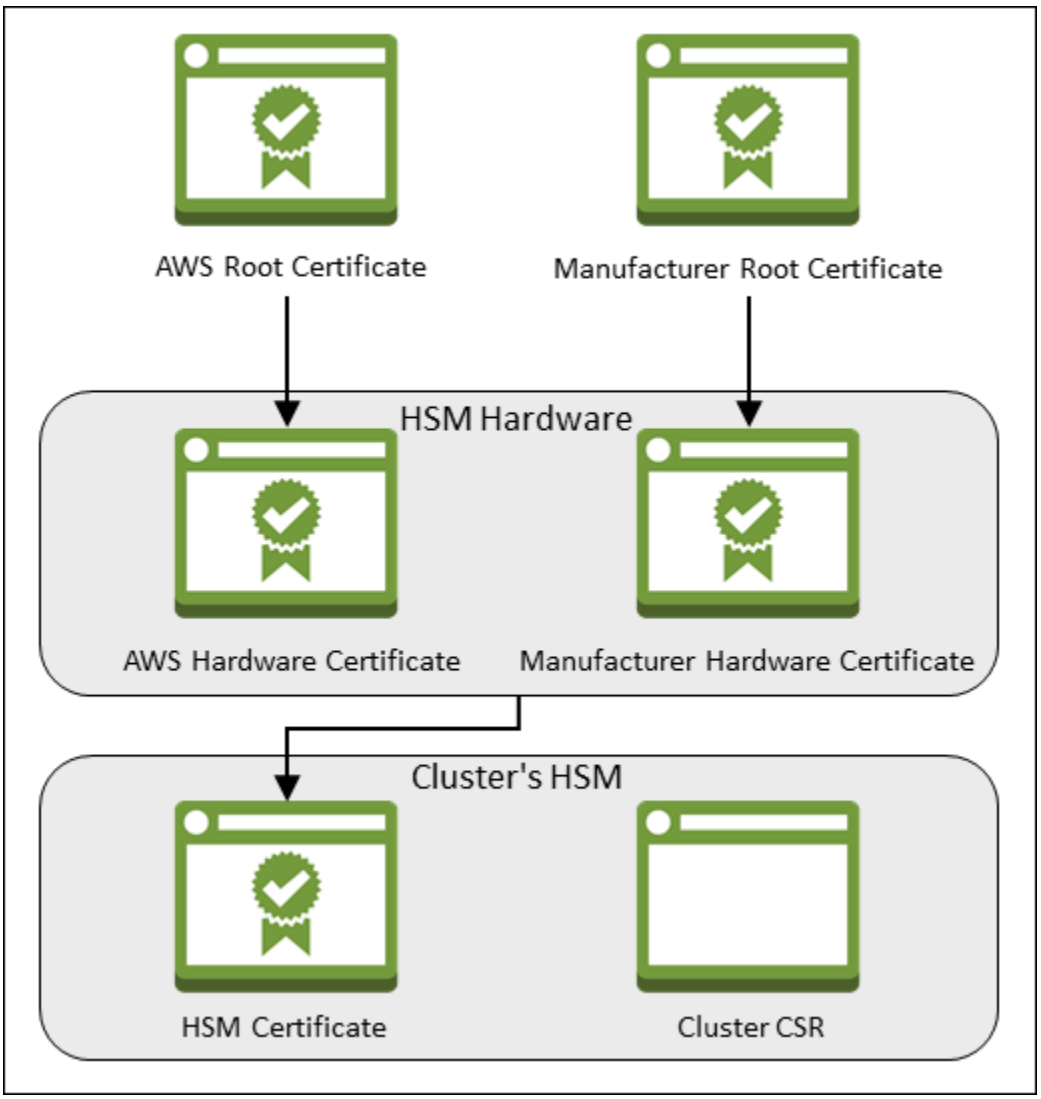
Tópicos

Visão geral

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

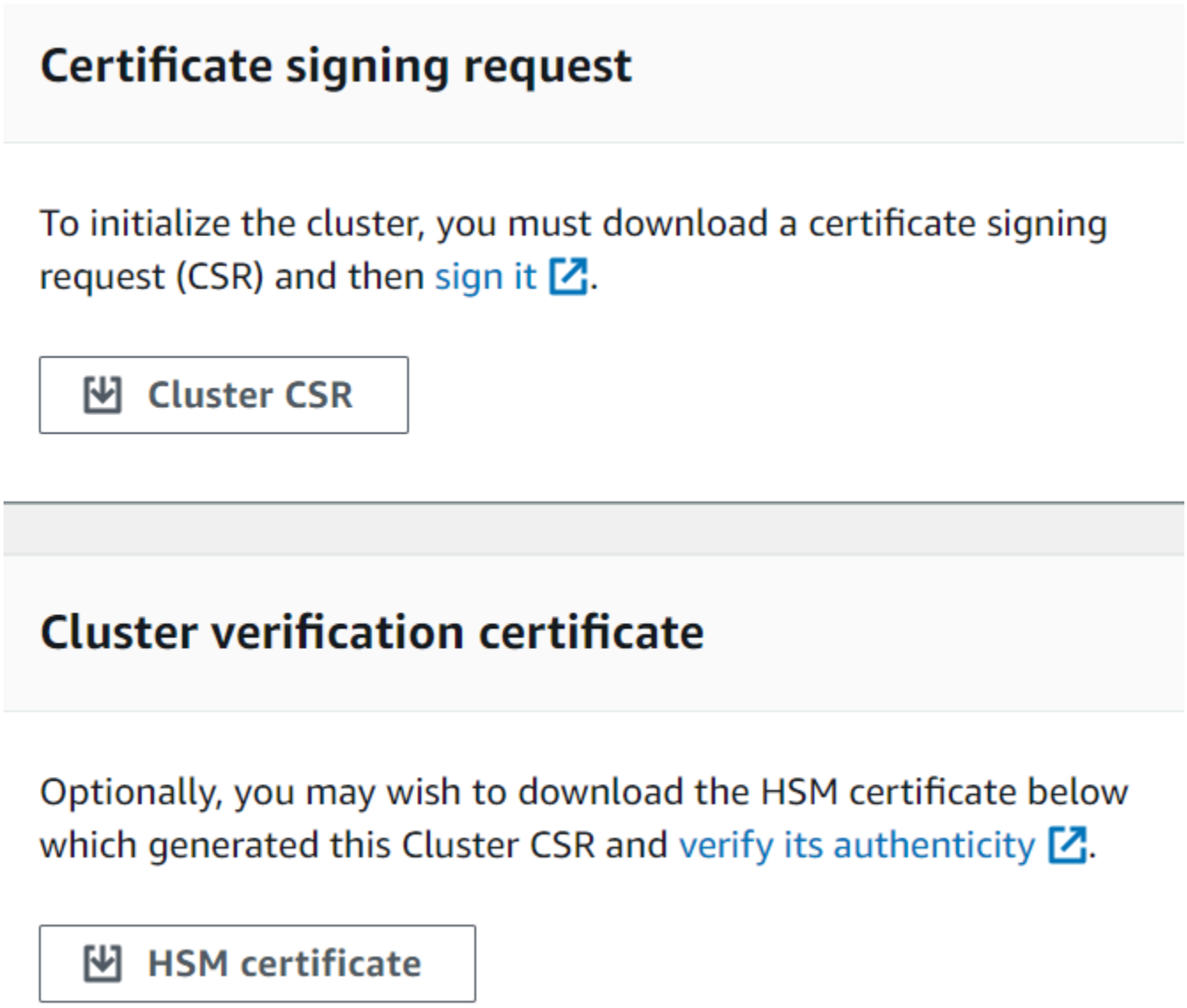
O primeiro HSM cria a CSR do cluster. Ao assinar a CSR do cluster, você reivindica o cluster. Em seguida, você pode usar a CSR assinada para inicializar o cluster.

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.

Para obter a CSR e certificados de HSM (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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.



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.

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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado 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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado 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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado 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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado anteriormente.

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

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.

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. Baixe o certificado raiz do fabricante certo para seu tipo de HSM:
   - certificado raiz do fabricante hsm1.medium: liquid_security_certificate.zip
   - certificado raiz do fabricante hsm2m.medium: liquid_security_certificate.zip

   > **Note**
   >
   > Para baixar cada certificado de sua página inicial, use os links a seguir:
   >
   > - Página inicial do certificado raiz do fabricante hsm1.medium
   > - Página inicial do certificado raiz do fabricante 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.

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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado 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 . Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado 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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado anteriormente.

   ```
   $ cat <cluster ID>_ManufacturerHardwareCertificate.crt \
         liquid_security_certificate.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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado anteriormente.

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

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.

Inicializar o cluster

Conclua as etapas nos tópicos a seguir para inicializar seu AWS CloudHSM cluster.

> **Note**
>
> Antes de inicializar o cluster, reveja o processo pelo qual você pode verificar a identidade e a autenticidade dos 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 os HSMs.

Tópicos

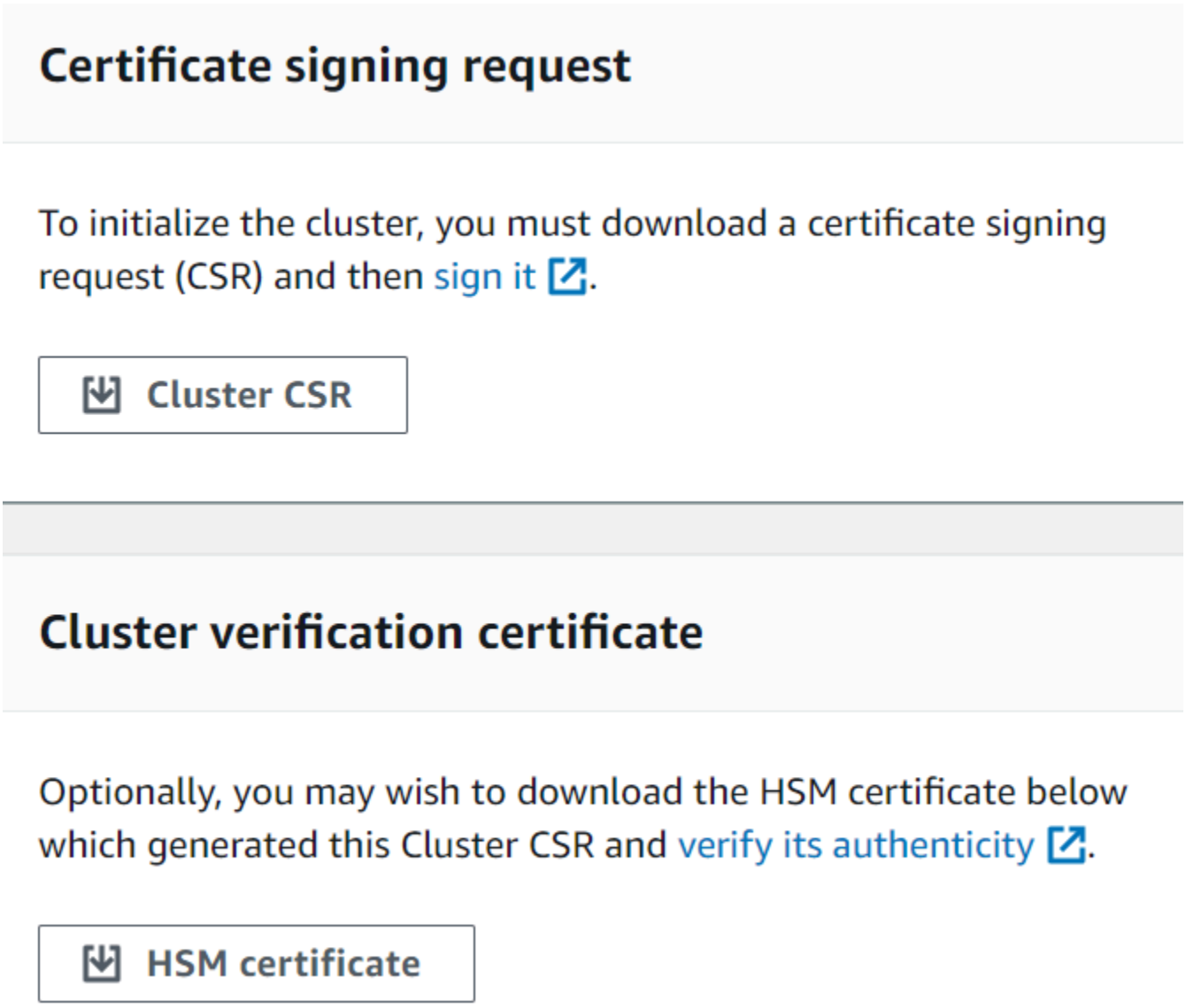
Obter a Solicitação de assinatura de certificado (CSR) do cluster

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.

> **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.

Para obter a CSR (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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.



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

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. Substitua o *<cluster ID>* pelo ID do cluster que tinha sido criado anteriormente.

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

Para obter o CSR (AWS CloudHSM API)

1. Envie uma solicitação DescribeClusters.
2. Extraia e salve a CSR da resposta.

Assinar a CSR

Atualmente, você deve criar um certificado de assinatura autoassinado e usá-lo para assinar a CSR do seu cluster. Você não precisa do AWS CLI para esta etapa e o shell não precisa estar associado à sua AWS conta. Para assinar a CSR, você deve fazer o seguinte:

1. Conclua a seção anterior (consulte Obter a Solicitação de assinatura de certificado (CSR) do cluster).
2. Crie uma chave privada.
3. Use a chave privada para criar um certificado de assinatura.
4. Assine a CSR do seu cluster.

Crie uma chave privada

> **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 os 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.

Use o comando a seguir para criar uma chave privada. Ao inicializar um AWS CloudHSM cluster, você deve usar o certificado RSA 2048 ou o certificado RSA 4096.

```
$ openssl genrsa -aes256 -out customerCA.key 2048
```

```
Generating RSA private key, 2048 bit long modulus
........+++
............+++
e is 65537 (0x10001)
Enter pass phrase for customerCA.key:
Verifying - Enter pass phrase for customerCA.key:
```

Usar a chave privada para criar um 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 criada na etapa anterior para criar um certificado de assinatura. 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 customerCA.key -out customerCA.crt
Enter pass phrase for customerCA.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 `customerCA.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.

Assinar a CSR do cluster

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 exemplo usa a chave privada e o certificado autoassinado criado na etapa anterior.

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

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

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.

Para inicializar um cluster (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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. Se tiver executado as etapas na seção anterior, selecione o arquivo denominado customerCA.crt.

c. Selecione Upload and initialize (Carregar e inicializar).

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. Se tiver executado as etapas na seção anterior, o certificado de assinatura foi salvo em um arquivo denominado customerCA.crt.

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

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.

- Seu certificado de assinatura.

Instalar e configurar a CLI do CloudHSM

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

Tarefas

- Instale as ferramentas da linha de AWS CloudHSM comando

Instale as ferramentas da linha de AWS CloudHSM comando

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 ter mais informações, consulte Iniciar uma instância do Amazon EC2..

Use os comandos a seguir para fazer download e instalar a CLI do CloudHSM.

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 arquitetura ARM64:

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

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

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 arquitetura ARM64:

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

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

CentOS 7 (7.8+)

CentOS 7 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
```

RHEL 7 (7.8+)

RHEL 7 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
```

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 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 na arquitetura ARM64:

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

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

Ubuntu 20.04 LTS

Ubuntu 20.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-cli_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u20.04_amd64.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 arquitetura ARM64:

```
$ 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 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 msiexec.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 msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /
quiet /norestart /log C:\client-install.txt' -Wait
```

Use o comando a seguir para configurar a CLI do CloudHSM.

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 IP 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.

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

Ativar o cluster

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 instância do EC2 que se conecta ao cluster (você cria o certificado de emissão ao inicializar o cluster).

Linux

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

Windows

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

Depois de colocar o certificado de emissão, instale a CLI do CloudHSM e execute o comando cluster activateem 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.

Ativar um cluster

1. Conecte-se à instância do cliente que você iniciou anteriormente. Para ter mais informações, consulte Iniciar uma instância do Amazon EC2.. 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

```
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.

Reconfigurar SSL com um novo certificado e chave privada (opcional)

AWS CloudHSM usa um certificado SSL para estabelecer uma conexão com um HSM. Uma chave padrão e certificado SSL são incluídos ao instalar o cliente. No entanto, você pode criar e usar seu próprio. Observe que você precisará do certificado autoassinado (*customerCA.crt*) que criou ao inicializar o cluster.

Basicamente, esse é um processo de duas etapas:

1. Primeiro, crie uma chave privada. Em seguida use a chave para criar uma solicitação de assinatura de certificado (CSR). Use o certificado de emissão, o certificado que você criou ao inicializar o cluster, para assinar a CSR.
2. Em seguida, você usa a ferramenta de configuração para copiar a chave e o certificado para os diretórios apropriados.

Crie uma chave, uma CSR e, em seguida, assine a CSR

As etapas são as mesmas para o Client SDK 3 ou Client SDK 5.

Para reconfigurar o SSL com um novo certificado e chave privada

1. Crie uma chave privada usando o seguinte comando OpenSSL:

```
openssl genrsa -out ssl-client.key 2048
Generating RSA private key, 2048 bit long modulus
........+++
............+++
e is 65537 (0x10001)
```

2. Use o comando OpenSSL a seguir para criar uma solicitação de assinatura de certificado (CSR). Você precisará responder uma série de perguntas sobre o certificado.

```
openssl req -new -sha256 -key ssl-client.key -out ssl-client.csr
```

```
Enter pass phrase for ssl-client.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) [XX]:
State or Province Name (full name) []:
Locality Name (eg, city) [Default City]:
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

3. Assine a CSR com o certificado *customerCA.crt* que você criou ao inicializar o cluster.

```
openssl x509 -req -days 3652 -in ssl-client.csr \
        -CA customerCA.crt \
        -CAkey customerCA.key \
        -CAcreateserial \
        -out ssl-client.crt
Signature ok
subject=/C=US/ST=WA/L=Seattle/O=Example Company/OU=sales
Getting CA Private Key
```

Ativar SSL personalizado para AWS CloudHSM

As etapas são diferentes para o Client SDK 3 ou Client SDK 5. Para obter mais informações sobre como trabalhar com a ferramenta de linha de comando de configuração, consulte ???.

Tópicos

- SSL personalizado para Client SDK 3

- SSL personalizado para Client SDK 5

SSL personalizado para Client SDK 3

Use a ferramenta de configuração do Client SDK 3 para ativar o SSL personalizado. Para obter mais informações sobre a ferramenta de configuração para o Client SDK 3, consulte ???.

Para usar um certificado e uma chave personalizados para autenticação mútua TLS cliente-servidor com o Client SDK 3 no Linux

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

```
sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
sudo /opt/cloudhsm/bin/configure --ssl \
 --pkey /opt/cloudhsm/etc/ssl-client.key \
 --cert /opt/cloudhsm/etc/ssl-client.crt
```

3. Adicione o certificado `customerCA.crt` ao armazenamento de confiança. Crie um hash do nome do certificado específico. Isso cria um índice para permitir que o certificado seja pesquisado por esse nome.

```
openssl x509 -in /opt/cloudhsm/etc/customerCA.crt -hash | head -n 1
1234abcd
```

   Crie um diretório.

```
mkdir /opt/cloudhsm/etc/certs
```

   Crie um arquivo que contenha o certificado com o nome de hash.

```
sudo cp /opt/cloudhsm/etc/customerCA.crt /opt/cloudhsm/etc/certs/1234abcd.0
```

SSL personalizado para Client SDK 5

Use qualquer ferramenta de configuração do Client SDK 5 para ativar o SSL personalizado. 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 TLS cliente-servidor com o Client SDK 5 no Linux

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` e. `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

OpenSSL Dynamic Engine

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

JCE provider

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-jce \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` e. `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-client.key
```

CloudHSM CLI

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-cli \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` e. `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-client.crt `
```

```
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

Para construir um aplicativo

Crie aplicativos e trabalhe com chaves usando AWS CloudHSM o.

Para começar a criar e usar chaves em seu novo cluster, você deve primeiro criar um usuário do módulo de segurança de hardware (HSM) com o CloudHSM Management Utility (CMU). 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.

Com os usuários do HSM instalados, você pode fazer login no HSM e criar e usar chaves com qualquer uma das seguintes 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 os provedores de CNG e KSP para usar com a Autoridade de Certificação (CA) do AWS CloudHSM Microsoft Windows Server
- Use os provedores de CNG e KSP para usar com a AWS CloudHSM Microsoft Sign Tool
- Use os provedores de CNG e KSP para descarga de TLS com o servidor Web do Internet Information Server (IIS)

Práticas recomendadas para AWS CloudHSM

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

Conteúdo

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 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 em 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.

Distribua seus HSMs em várias zonas de disponibilidade: considere como seu serviço será capaz de operar durante uma interrupção na zona de disponibilidade. AWS recomenda que você distribua seus HSMs pelo maior número possível de zonas de disponibilidade. Para um cluster com três HSMs, você deve distribuir os HSMs em 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 distribuídas em todas as zonas de disponibilidade de uma região.

Acesso seguro ao cluster

Use sub-redes privadas para limitar o acesso à sua instância: execute seus HSMs e instâncias de clientes nas sub-redes privadas da sua VPC. Isso limita o acesso aos seus HSMs do mundo exterior.

Use VPC endpoints para acessar as APIs: o plano de AWS CloudHSM dados foi projetado para operar sem a necessidade de acesso à Internet ou às APIs da AWS. 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 mais informações.

Reconfigure o SSL para proteger a comunicação cliente-servidor: AWS CloudHSM usa o TLS para estabelecer uma conexão com seu HSM. Depois de inicializar seu cluster, você pode substituir o certificado TLS padrão e a chave usados para estabelecer a conexão TLS externa. Para ter mais informações, consulte Melhore a segurança do seu servidor web com o descarregamento de SSL/TLS em AWS CloudHSM.

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 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 os HSMs forem novamente necessários, você poderá restaurá-los a partir de um backup. Se, por exemplo, você tiver uma workload exigindo que você assine o código uma vez por mês, especificamente no último dia do mês, você pode aumentar a escala do seu cluster verticalmente antes, reduzi-lo excluindo seus 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 dos 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 entre AWS regiões

Gerenciamento de usuários do HSM

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 da IAM que têm uma política baseada em identidade com as permissões apropriadas podem criar HSMs interagindo com recursos 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, consulteGerenciando 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 Client SDK 5 são sinônimos de agentes criptográficos (CoS) no Client SDK 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:

- compartilhamento de chaves
- descompartilhar chave

Gerenciamento de chaves HSM

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. HSMs extras em seu cluster não aumentarão a throughput das solicitações dessa chave. Se você usar uma chave de token para o mesmo aplicativo, suas solicitações terão a carga balanceada em todos os HSMs disponíveis em seu cluster. Para ter mais informações, consulte Configurações de sincronização e durabilidade de teclas em AWS CloudHSM.

Gerencie os principais limites de armazenamento

Os 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 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 clusters com o provedor JCE
- Conectando-se a vários slots com PKCS#11

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 encapsulamento de chaves em seu aplicativo, deverá utilizar o encapsulamento de chaves confiável para limitar a capacidade dos usuários do HSM de encapsular/desencapsular somente chaves que tenham sido explicitamente marcadas como confiáveis por um administrador. Para obter mais informações, consulte tópicos sobre encapsulamento de chaves confiáveis em Gerenciando chaves em AWS CloudHSM.

Recursos relacionados

- Funções de agrupamento e desagrupamento
- Funções de criptografia para JCE
- Atributos de chave Java compatíveis
- Atributos de chave da CLI do CloudHSM

Integração de aplicações

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 bookstrap. 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. Para obter mais detalhes, consulte 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 CLI do CloudHSM: você pode se autenticar com a CLI do CloudHSM usando seu modo de comando único ou o modo interativo. Use o comando login 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, consulteModo de comando único. AWS CloudHSM recomenda armazenar com segurança suas credenciais do HSM quando não estiverem sendo usadas pelo seu aplicativo.

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.

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, consulteForneça credenciais ao provedor JCE.

Autenticar com OpenSSL: com o OpenSSL Dynamic Engine, você fornece credenciais por meio de variáveis de ambiente. AWS CloudHSM recomenda armazenar com segurança suas credenciais do HSM quando não estiverem sendo usadas pelo seu aplicativo. 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 Instalação do Mecanismo dinâmico do OpenSSL.

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 dos seguintes SDKs do cliente:

- 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 teclas (como findKey elista de chaves) 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 HSMs disponíveis 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 do mesmo aplicativo: o provedor JCE, a biblioteca PKCS #11 e a CLI do CloudHSM no Client SDK 5 oferecem suporte à conexão com vários clusters do mesmo aplicativo. 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:

- Conectando-se a vários slots com PKCS#11
- Conectando-se a vários clusters com o provedor JCE
- Conectando-se a vários clusters com o CloudHSM CLI

Restaurar um cluster a partir de um backup: você pode criar um novo cluster a partir do backup de um cluster existente. Para ter mais informações, consulte Gerenciando AWS CloudHSM backups.

Monitorar

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 Trabalhar com logs do Client SDK.

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 mantê-la e visualizá-la.

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 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
- HSMs não íntegros para escalar seu cluster até que AWS CloudHSM possam resolver o problema

Para obter mais detalhes, consulte Trabalhando com Amazon CloudWatch Logs e AWS CloudHSM Audit Logs.

Gerenciando AWS CloudHSM clusters

Você pode gerenciar seus AWS CloudHSM clusters a partir do AWS CloudHSM console ou de um dos AWS SDKs ou ferramentas de linha de comando. Para obter mais informações, consulte os tópicos a seguir.

Para criar um cluster, consulte Conceitos básicos.

Arquitetura do cluster

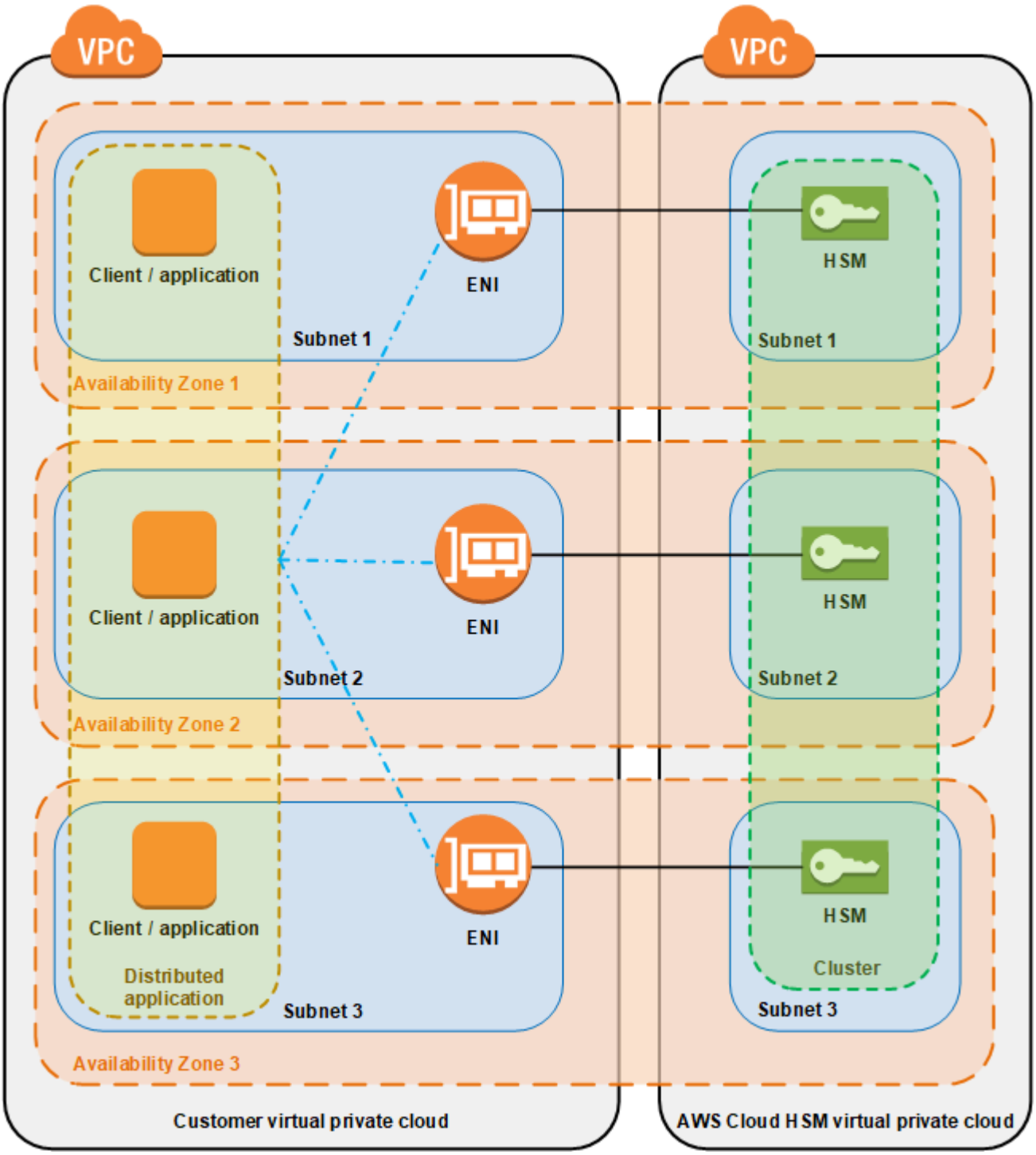
Ao criar um cluster, você especifica uma Amazon Virtual Private Cloud (VPC) em sua AWS conta e uma ou mais sub-redes nessa VPC. Recomendamos que você crie uma sub-rede em cada Zona de Disponibilidade (AZ) na AWS região escolhida. Você pode criar sub-redes privadas ao criar uma VPC. Para saber mais, consulte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

Toda vez que criar um HSM, especifique o cluster e a zona de disponibilidade para o HSM. Ao colocar os HSMs em diferentes zonas de disponibilidade, você alcançará redundância e alta disponibilidade caso uma zona de disponibilidade não esteja disponível.

Ao criar um HSM, AWS CloudHSM coloca uma interface de rede elástica (ENI) na sub-rede especificada em sua conta. AWS 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, o cliente é instalado em instâncias do Amazon EC2, conhecidas como instâncias cliente, que residem na mesma VPC que ENIs de HSM, conforme mostrado na figura a seguir. No entanto, isto não é tecnicamente necessário; é possível instalar o cliente em qualquer computador compatível, desde que ele possa ser conectado a ENIs de HSM. O cliente se comunica com HSMs individuais no cluster por meio dos ENIs respectivos.

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



Sincronização do cluster

Em um AWS CloudHSM cluster, AWS CloudHSM mantém as chaves dos HSMs individuais sincronizadas. Você não precisa fazer nada para sincronizar as chaves nos 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 ter 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.

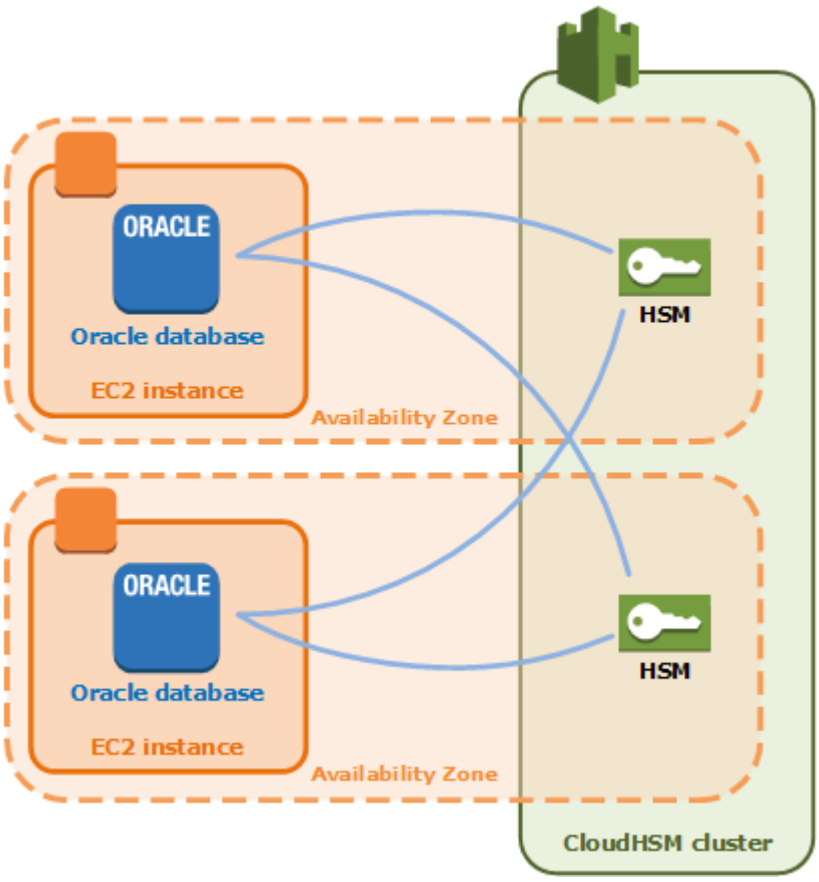
Se os HSMs em um cluster ficarem fora de sincronização, eles AWS CloudHSM serão ressincronizados automaticamente. Para habilitar isso, AWS CloudHSM usa as credenciais do usuário do equipamento. Esse usuário existe em todos os HSMs fornecidos por AWS CloudHSM e tem permissões limitadas. Ele pode obter 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.

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. Balanceamento de carga significa que o cliente do AWS CloudHSM distribui as operações criptográficas por todos os HSM no cluster com base na capacidade de processamento adicional de cada HSM.

Ao criar os 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 os dois HSMs para que fiquem imediatamente acessíveis e redundantes.



AWS CloudHSM modos de cluster e tipos de HSM

AWS CloudHSM oferece dois modos de cluster: FIPS e não FIPS. AWS CloudHSM também oferece dois tipos de HSM: hsm1.medium e hsm2m.medium. Analise os detalhes nesta página antes de decidir qual modo de cluster e tipo de HSM é adequado às suas necessidades.

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

Para ver o modo e o tipo de HSM do seu cluster, use o comando describe-clusters.

Modos de cluster

AWS CloudHSM oferece clusters em dois modos: FIPS e não FIPS. No modo FIPS, somente chaves e algoritmos aprovados pelo Federal Information Processing Standard (FIPS) podem ser

usados. O modo não FIPS oferece todas as chaves e algoritmos suportados AWS CloudHSM, independentemente da aprovação do FIPS.

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 HSM	Disponível com hsm1.medium.	Disponível com hsm2m.medium.
Compatibilidade retroativa	Só pode ser usado para fazer backup de clusters de restauração no modo FIPS.	Só pode ser usado para fazer backup de clusters de restauração no modo não FIPS.
Seleção de chaves	Suporta AWS CloudHSM chaves [1] aprovadas pelo FIPS.	Suporta AWS CloudHSM chaves aprovadas pelo FIPS e não aprovadas pelo FIPS.
Algoritmos	Suporta AWS CloudHSM algoritmos [1] aprovados pelo FIPS.	Suporta AWS CloudHSM algoritmos aprovados pelo FIPS e não aprovados pelo FIPS.
Certificação	Compatível com FIPS 140-2, PCI PIN e PCI-3DS.	

[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 após sua criação, portanto, certifique-se de selecionar o modo certo para suas necessidades.

Tipos de HSM

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 com o modo de cluster	Disponível para clusters no modo FIPS.	Atualmente disponível para clusters no modo não FIPS.
Compatibilidade retroativa	Só pode ser usado para fazer backup e restauração em clusters hsm1.medium.	Só pode ser usado para fazer backup e restaurar clusters hsm2m.medium.
Capacidade chave	3.300 por cluster.	Total de 16.666 chaves, com chaves assimétricas com um máximo de 3.333 por cluster.
Client SDKs	Compatível com todos os Client SDKs.	Compatível com todos os SDKs do cliente, exceto os provedores de CNG e KSP.
Versões do Client SDK	Compatível com o SDK versão 3.1.0 e posterior.	Compatível com o SDK do cliente versão 5.12.0 e posterior.
Disponibilidade de regiões	Disponível em todas as regiões em que o CloudHSM está disponível.	Disponível em um número limitado de regiões, com suporte adicional em breve. Para ver as regiões em que esse tipo de HSM está disponível, consulte a calculadora AWS CloudHSM de preços.
Desempenho	Para ver o desempenho de cada tipo de HSM, consulte AWS CloudHSM Desempenho.	
Certificação	Compatível com FIPS 140-2, PCI DSS, PCI PIN, SOC2 e PCI-3DS.	Compatível com PCI DSS.

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

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 SDK do cliente 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.

```
"C:\Program Files\Amazon\CloudHSM\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>
```

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.

```
"C:\Program Files\Amazon\CloudHSM\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.

```
"C:\Program Files\Amazon\CloudHSM\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 Client SDK se conecta, ele recupera os endereços IP de todos os HSMs adicionais 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/home.
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, na guia HSMs, escolha um dos endereços IP listados em Endereço IP ENI.

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`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

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.

```
"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>
```

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.

```
"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 IP 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.

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

> **Note**
>
> você pode usar o parâmetro `--cluster-id` no lugar de `-a <HSM_IP_ADDRESSES>`. Para ver os requisitos de uso de `--cluster-id`, consulte 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 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 configure para especificar o endereço IP de um HSM no seu cluster.

```
C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe -a <HSM IP address>
```

Para obter mais informações sobre configuração, consulte ???.

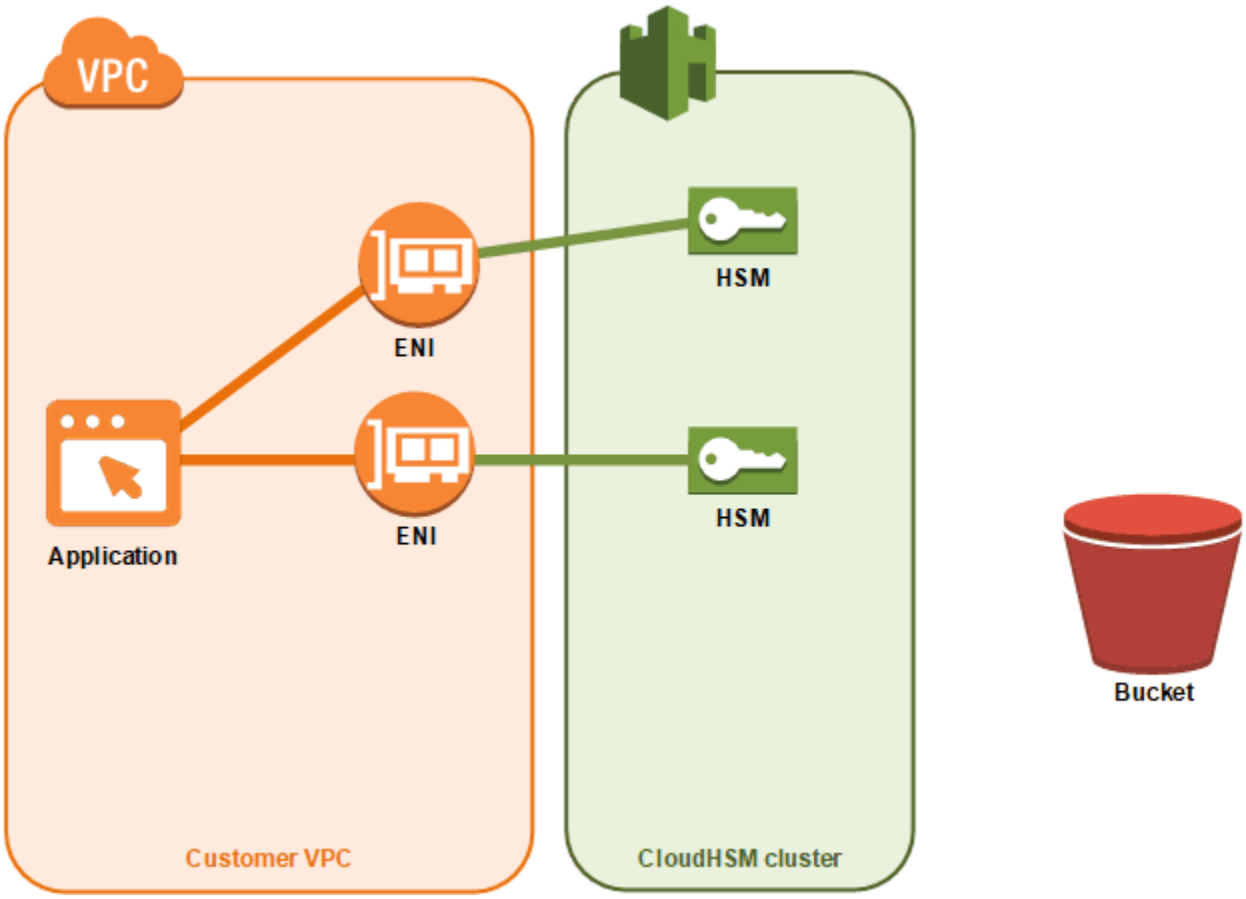
Adicionar ou remover HSMs em um cluster AWS CloudHSM

Para ampliar ou reduzir seu AWS CloudHSM cluster, adicione ou remova HSMs usando o AWS CloudHSM console ou um dos AWS SDKs ou ferramentas de 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 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 ter 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. Os HSMs existentes no cluster notificam 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/home.

2. Escolha um cluster para o HSM que você está adicionando.
3. Na guia HSMs, escolha Create HSM.
4. Escolha uma zona de disponibilidade (AZ) para o HSM que está sendo criado. Em seguida, selecione 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"
    }
}
```

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

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/home.
2. Escolha o cluster que contém o HSM que você está removendo.
3. Na guia HSMs, escolha o HSM que você está removendo. Em seguida, escolha Delete HSM.
4. Confirme que você deseja excluir o HSM. Em seguida, selecione 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 poder excluir um cluster, é necessário remover todos os HSMs do cluster. Para ter mais informações, consulte Removendo um HSM.

Depois de remover todos os 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)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
2. Escolha o cluster que você está excluindo. Em seguida, escolha Delete cluster.
3. 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 Gerenciar backups.

Crie clusters a partir de backups (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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.
4. Na seção Cluster source, faça o seguinte:
   a. Escolha Restore cluster from existing backup.
   b. Escolha o backup que você está restaurando.
5. Selecione Next: Review (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 ter 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 nesse cluster que contenha os mesmos usuários, material chave, certificados, configuração e políticas que estavam no backup que você restaurou, adicione um HSM ao cluster.

Crie clusters a partir 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, os IDs das sub-redes em que você planeja criar HSMs e o ID de backup do backup que você está restaurando.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm1.medium \
                                --subnet-ids <subnet ID 1> <subnet ID 2> <subnet ID
 N> \
                                --source-backup-id <backup ID>
{
    "Cluster": {
        "HsmType": "hsm1.medium",
        "VpcId": "vpc-641d3c0d",
        "Hsms": [],
        "State": "CREATE_IN_PROGRESS",
        "SourceBackupId": "backup-rtq2dwi2gq6",
        "BackupPolicy": "DEFAULT",
        "BackupRetentionPolicy": {
            "Type": "DAYS",
            "Value": 90
         },
        "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

Gerenciando AWS CloudHSM backups

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

Trabalhar com backups

Quando você adiciona um HSM a um cluster que anteriormente continha um ou mais HSMs ativos, o serviço restaura o backup mais recente para o novo HSM. Use backups para gerenciar HSMs que você usa 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 e restaurar um backup”.

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 Criação de clusters a partir 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 e restaurar um backup

Depois de excluir um backup, o serviço retém o backup por sete dias, período durante o qual você pode restaurá-lo. Após o período de sete dias, você não poderá mais restaurar o backup. Para obter mais informações sobre gerenciar backups, consulte Gerenciar backups.

Excluir e restaurar backups (console)

Para excluir um backup (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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 restaurar um backup (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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 no `PENDING_DELETE` estado a ser restaurado.
5. Para restaurar o backup selecionado, escolha Ações, Restaurar.

Excluir e restaurar backups (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"
    }
```

```
}
```

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"
    }
}
```

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,
            "DeleteTimestap": 1536339805.522,
            "HsmType": "hsm2m.medium",
            "Mode": "NON_FIPS",
            "NeverExpires": false,
            "TagList": []
        }
}
```

Excluir e restaurar backups (AWS CloudHSM API)

Consulte os tópicos a seguir para saber como excluir e restaurar backups usando a API.

- DeleteBackup
- RestoreBackup

Configurando a política de retenção de AWS CloudHSM backup

Com a exceção dos clusters criados antes de 18 de novembro de 2020, a política padrão de retenção de backup para clusters é de 90 dias. Você pode definir esse período para qualquer número entre 7 e 379 dias. AWS CloudHSM não exclui o último backup de um cluster. Para obter mais informações sobre gerenciar backups, consulte Gerenciar backups.

Noções básicas da política de retenção de 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 de um cluster.

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.

Isenção de cluster existente

AWS CloudHSM lançou a retenção gerenciada de backup em 18 de novembro de 2020. 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).

> **Note**
>
> Você pode desativar totalmente a retenção gerenciada de backup entrando em contato com o suporte (https://aws.amazon.com/support).

Configurar a retenção de backup (console)

Para configurar a retenção de backup (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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/home.
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.

Configurar a retenção de backup (AWS CLI)

Verifique o status de um backup ou encontre sua ID usando o describe-backups comando do 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
   }
}
```

Configurar retenção de backup (AWS CloudHSM API)

Consulte os tópicos a seguir para saber como gerenciar a retenção de backups usando a API.

- ModifyCluster
- ModifyBackupAttributes

Copiar backups entre AWS regiões

Você pode copiar backups 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 Gerenciar backups ou Criação de clusters a partir 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 ter 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 ter mais informações, consulte Gerenciamento de clusters do .
- Os backups não podem ser copiados entre regiões AWS restritas e regiões padrão. Os backups podem ser copiados entre as regiões AWS GovCloud (Leste dos EUA) e AWS GovCloud (Oeste dos EUA).

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/home.
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

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

- 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. Você não pode compartilhar um backup que tenha sido compartilhado com você.
- Para compartilhar um backup, ele deve estar no estado PRONTO.
- Para compartilhar um backup com sua organização ou unidade organizacional em AWS Organizations, você deve habilitar o compartilhamento com AWS Organizations. Para obter mais informações, consulte Habilitar o compartilhamento com o AWS Organizations no Guia do usuário do AWS RAM .

Compartilhando 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, você deve 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 consumidores com os quais compartilhá-los. Ao compartilhar um backup usando o console do CloudHSM, você o adiciona a um compartilhamento de recursos existente. Para adicionar o backup a um novo compartilhamento de recursos, primeiro você deve criar o compartilhamento de recursos usando o AWS RAM console.

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 terão acesso ao backup compartilhado após aceitarem 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 atributos 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 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.

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

```
    "Effect":"Allow",
    "Principal":{
      "AWS":"<consumer-aws-account-id-or-user>"
    },
    "Action":[
      "cloudhsm:CreateCluster",
      "cloudhsm:DescribeBackups"],
    "Resource":"<arn-of-backup-to-share>"
  }]
}
```

2. Atualize `policy.json` com o ARN de backup e os identificadores com os quais compartilhar. O exemplo a seguir concede acesso somente de leitura ao usuário raiz da AWS conta identificada por 123456789012.

```
{
  "Version":"2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Principal":{
      "AWS": [
        "account-id"
      ]
    },
    "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 valor *< resource-share-arn >* para usar 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-sharescomando.

```
$ aws ram get-resource-shares --resource-owner SELF --resource-share-
arns <resource-share-arn>
```

Quando a política é promovida, a `featureSet` listada na resposta é`STANDARD`. Isso também significa que o backup pode ser descrito pelas novas contas na política.

Cancelar o compartilhamento de um backup compartilhado

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, você deve 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 CloudHSM)

Use o comando delete-resource-policy.

```
$ aws cloudhsmv2 delete-resource-policy --resource-arn <resource-arn>
```

Identificação de um backup compartilhado

Os consumidores podem identificar um backup compartilhado com eles usando o console do CloudHSM e. AWS CLI

Para identificar backups compartilhados com você usando o console do CloudHSM

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.
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 `--shared` parâmetro 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 consumidores

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.

Recursos de marcação AWS CloudHSM

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 console do AWS CloudHSM ou um dos SDKs da AWS ou das ferramentas de linha de comando para adicionar, atualizar, listar e remover tags.

Tópicos

- Adicionar ou atualizar tags
- Listar tags
- Remover tags

Adicionar ou atualizar tags

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/home.
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

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/home.
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

É possível remover tags de um cluster usando o console do AWS CloudHSM, a AWS CLI ou a API do AWS CloudHSM .

Para remover tags (console)

1. Abra o AWS CloudHSM console em https://console.aws.amazon.com/cloudhsm/home.

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 UntagResourcesolicitação na AWS CloudHSM API, especificando o ID do cluster e as tags que você está removendo.

Gerenciando usuários e chaves do HSM no AWS CloudHSM

Antes de usar seu AWS CloudHSM cluster para processamento de criptografia, você deve criar usuários e chaves nos HSMs do seu cluster. Consulte os seguintes tópicos para obter mais informações sobre como gerenciar usuários e chaves do HSM AWS CloudHSM. Você também pode aprender a usar a autenticação de quorum (também conhecida como controle de acesso M de N).

Tópicos

- Gerenciando usuários do HSM em AWS CloudHSM
- Gerenciando chaves em AWS CloudHSM
- Gerenciando clusters clonados

Gerenciando usuários do HSM em AWS CloudHSM

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.

Tópicos

- Gerenciando chaves com o CloudHSM CLI
- Gerenciar usuários do HSM com o CloudHSM Management Utility (CMU)

Gerenciando chaves com o CloudHSM CLI

Use as ferramentas de linha de comando CLI do CloudHSM para criar e gerenciar os usuários no seu HSM com o SDK mais recente.

Tópicos

- Noções básicas sobre usuários do HSM
- Tabela de permissões de usuário do HSM
- Usar a CLI do CloudHSM para gerenciar usuários
- Usar a CLI do CloudHSM para gerenciar a MFA

- Como usar o CLI do CloudHSM para gerenciar a autenticação de quórum (controle de acesso M ou N)

Noções básicas sobre usuários do HSM

A maioria das operações executadas no HSM exigem as credenciais de um usuário do 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 HSMs interagindo com recursos 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
- Usuário de criptografia (CU)
- 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 ativar um cluster.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.

Usuário de criptografia (CU)

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 Tabela de permissões de usuário do HSM.

Usuário de dispositivo (AU)

O usuário do equipamento (AU) pode realizar operações de clonagem e sincronização nos HSMs do seu cluster. AWS CloudHSM usa a AU para sincronizar os HSMs em um AWS CloudHSM cluster. A AU existe em todos os HSMs fornecidos por AWS CloudHSM, e tem permissões limitadas. Para obter mais informações, consulte Tabela de permissões de usuário do HSM.

AWS não pode realizar nenhuma operação em seus 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

A tabela a seguir lista as operações do HSM classificadas pelo tipo de usuário ou sessão do HSM que pode realizar a operação.

	Administrador	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Obter informaçõ es[1] básicas do cluster	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)
Alterar a própria senha	Yes (Sim)	Yes (Sim)	Yes (Sim)	Não aplicável

	Administrador	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Alterar a senha de qualquer usuário	 Sim	 Não	 No (Não)	 No (Não)
Adicionar , remover usuários	 Sim	 Não	 No (Não)	 No (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)	 Não
Funções de gerenciamento de chaves[5]	 No (Não)	 Sim	 Não	 No (Não)
Criptografar, descriptografar	 No (Não)	 Sim	 Não	 No (Não)
Assinar, verificar	 No (Não)	 Sim	 Não	 No (Não)

	Administrador	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Gerar resumos e HMACs	 No (Não)	 Sim	 Não	 No (Não)

- [1] As informações básicas de cluster incluem o número de HSMs no cluster e cada endereço IP do HSM, o modelo, o número de série, o ID do dispositivo, o 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 arquivos de resumo para compreender o status de sincronização de 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 sincronizar os HSMs em um cluster.
- [4] As funções de gerenciamento de chaves incluem criar, excluir, encapsular, desencapsular e modificar os atributos das chaves.

Usar a CLI do CloudHSM para gerenciar usuários

Este tópico fornece step-by-step instruções sobre como gerenciar usuários do módulo de segurança de hardware (HSM) com a CLI do CloudHSM. Para obter mais informações sobre a CLI do CloudHSM, consulte CLI do CloudHSM e Uso da CLI do CloudHSM.

Seções

- Noções básicas sobre gerenciamento de usuários do HSM com a CLI do CloudHSM
- Fazer o download da CLI do CloudHSM
- Como gerenciar usuários do HSM com o CloudHSM CLI

Noções básicas sobre gerenciamento de usuários do HSM com a CLI do CloudHSM

Para gerenciar os usuários do HSM, faça 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.

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. Conceitos básicos com a Interface da Linha de Comando da CloudHSM (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/home.
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, na guia HSMs, escolha um dos endereços IP listados em Endereço IP ENI.

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`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
```

```
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

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.

Como gerenciar usuários do HSM com o CloudHSM CLI

Esta seção inclui comandos básicos para gerenciar usuários do HSM com a CLI do CloudHSM.

> **Note**
>
> Observação: os comandos de usuário da CLI do CloudHSM estão listados na referência de comandos do usuário da CLI do CloudHSM

Tópicos

- Para criar um administrador
- Para criar um usuário de criptografia
- Para listar todos os usuários do HSM no cluster
- Para alterar as senhas de usuário do HSM
- Excluir usuários do HSM

Para criar um administrador

Siga estas etapas para criar um administrador.

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

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

Windows

```
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": "admin",
    "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.

Para criar um usuário de criptografia

Para criar um usuário, siga as etapas.

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

Linux

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

Windows

```
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": "admin",
    "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.

Para listar todos os usuários do HSM no cluster

Use o comando user list para listar todos os usuários no cluster. Você não precisa fazer login para executaruser 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

```
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.

Para alterar as senhas de usuário do HSM

Use o user change-password comando para alterar uma senha.

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

   ```
   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": "admin1",
    "role": "admin"
  }
}
```

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

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

2. Use o comando login e faça login como um administrador.

```
aws-cloudhsm > login --username <USERNAME> --role admin
```

3. Digite a senha do administrador.

```
Enter password:
{
```

```
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
}
```

4. 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>
```

5. Digite a nova senha.
6. 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

Use user delete para excluir um usuário. 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

   ```
   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": "admin",
    "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.

Usar a CLI do CloudHSM para gerenciar a MFA

Para aumentar a segurança, você pode configurar a autenticação multifator (MFA) para ajudar a proteger o cluster. Para obter mais informações, consulte os tópicos abaixo:

Tópicos

- Noções básicas sobre MFA para usuários de HSM
- Trabalhar com MFA para usuários de HSM

Noções básicas sobre MFA para usuários de HSM

Ao fazer login em um cluster com uma conta de mde usuário de 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 o MFA para um usuário, consulte Configurar o MFA para a CLI do CloudHSM

Autenticação de quórum e MFA

O cluster usa a mesma chave para autenticação de quórum e MFA. Isso significa que um usuário com MFA habilitado está efetivamente registrado para controle de acesso MoFN ou Quorum. 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 quorum, registre esse usuário como um Quorum (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 Gerenciamento de quórum (M de N).

Trabalhar com MFA para usuários de HSM

Este tópico fornece informações e instruções para usar a CLI do CloudHSM para gerenciar a autenticação multifator (MFA). Para obter mais informações sobre a CLI do CloudHSM, consulte Interface de linha de comando (CLI) do CloudHSM.

Tópicos

- Requisitos de pares de chaves MFA
- Configurar o MFA para a CLI do CloudHSM
- Crie usuários com MFA habilitado
- Faça login com usuários com o MFA ativado
- Rotacione as chaves para usuários com MFA habilitado

- Cancele o registro de uma chave pública de MFA para usuários administradores quando a chave pública de MFA for registrada
- Referência do arquivo de token

Para obter mais informações sobre como trabalhar com usuários do HSM, consulte Interface de linha de comando (CLI) do CloudHSM.

Requisitos de pares de chaves MFA

Para ativar MFA para um usuário de HSM, 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 Autenticação de quórum e MFA.

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 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. Usando a CLI do CloudHSM, faça login na sua conta de usuário.

```
$ cloudhsm-cli interactive
aws-cloudhsm > login --username admin --role admin --cluster-id <cluster ID>
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin",
    "role": "admin"
  }
}
```

3. 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:
```

4. 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/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=",
      "signed": ""
    },
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
```

```
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
      "signed": ""
    }
  ]
}
```

5. 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/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=" > 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
```

6. 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
```

7. Agora, você tem assinaturas binárias dos tokens. Você precisa codificá-los usando base64 e colocá-los de volta 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
```

8. Finalmente, você pode copiar e colar os valores de base64 de volta em seu arquivo de token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQ
    }
  ]
}
```

9. 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

Siga estas etapas para criar usuários com o MFA ativado.

1. Use a CLI do CloudHSM para fazer login no HSM como administrador.
2. Use o user createcomando 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 usuários com o MFA ativado

Siga estas etapas para fazer login de usuários com o MFA ativado.

1. Use o login mfa-token-signcomando 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 pares de tokens não assinados/assinados, 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/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=",
      "signed": ""
    },
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
      "signed": ""
    }
  ]
}
```

4. 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/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=" > 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
```

5. 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
```

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": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQ
    }
  ]
}
```

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

Siga estas etapas para alternar chaves para usuários com o MFA ativado.

<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 chaves públicas/privadas RSA que você deseja usar para alternar a chave pública atualmente registrada.

   ```
   $cat unsigned-tokens.json
   {
     "version": "2.0",
     "tokens": [
       {
         "unsigned": "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=",
         "signed": ""
       },
   ```

```
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
      "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/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=" > 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
```

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": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQ
    }
  ]
}
```

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"
},
```

Cancele o registro de uma chave pública de MFA para usuários administradores quando a chave pública de MFA for registrada

Siga estas etapas para cancelar o registro de uma chave pública de MFA para usuários administradores 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 do arquivo de token

O arquivo de token gerado ao registrar uma chave pública de MFA ou ao tentar fazer login usando o MFA consiste no seguinte:

- Tokens: uma matriz de pares de tokens não assinados/assinados codificados em base64 na forma de literais de objetos JSON.
- Não assinado: um token codificado em base64 e com hash SHA256 do approval_data.
- 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": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/TK0PVaxLN42X
+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37+j/
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
```

```
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQdErI
    }
  ]
}
```


Como usar o CLI do CloudHSM para gerenciar a autenticação de quórum (controle de acesso M ou N)

Os HSMs em seu AWS CloudHSM cluster oferecem 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 administrador: criar e excluir usuários do HSM e alterar a senha de um usuário do HSM diferente. Para ter mais informações, consulte Usar a autenticação de quórum para administradores.

Os seguintes tópicos fornecem mais informações sobre a autenticação de quorum no AWS CloudHSM.

Tópicos

- Visão geral da autenticação de quórum com a estratégia de assinatura com token
- Detalhes adicionais sobre a autenticação de quorum
- Nomes e tipos de serviços que oferecem suporte à autenticação de quórum
- Usar a autenticação de quórum para administradores: configuração inicial

- Usar a autenticação de quórum para administradores
- Altere o valor mínimo do quórum para administradores

Visão geral da autenticação de quórum com a estratégia de assinatura com token

As etapas a seguir resumem os processos de autenticação de quorum. Para conhecer as etapas e ferramentas específicas, consulte Usar a autenticação de quórum para administradores.

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 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.

Detalhes adicionais sobre a autenticação de quorum

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.
- Se MFA estiver ativada, o cluster usa a mesma chave para autenticação de quórum e para autenticação multifator (MFA). Para obter mais informações sobre o uso da autenticação de quórum e da 2FA, consulte Como usar a CLI do CloudHSM para gerenciar a MFA.
- Cada HSM pode conter apenas um token por serviço por vez.

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.

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

Usar a autenticação de quórum para administradores: configuração inicial

Os tópicos a seguir descrevem as etapas que você deve concluir para configurar seu hardware security module (HSM) de forma que os 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 Usar a autenticação de quórum para administradores.

Tópicos

- Pré-requisitos
- Criar e registrar uma chave para assinatura
- Definir o valor mínimo do quorum no HSM

Pré-requisitos

Para entender esse exemplo, você deve estar familiarizado com o CloudHSM CLI. Neste exemplo, o AWS CloudHSM cluster tem dois HSMs, cada um com os mesmos administradores, conforme mostrado na saída a seguir do comando. user list Para obter mais informações sobre como criar usuários, consulte Uso da CLI do CloudHSM.

```
aws-cloudhsm>user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "admin2",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "admin3",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
```

```
        "username": "admin4",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

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.

Example 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, substitua *<admin.key>* pelo nome do arquivo onde você deseja armazenar a chave.

```
$ openssl genrsa -out <admin.key> -aes256 2048
```

```
Generating RSA private key, 2048 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
Enter pass phrase for admin.key:
Verifying - Enter pass phrase for admin.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 admin.key -outform PEM -pubout -out admin1.pub
Enter pass phrase for admin.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 de registro

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

   Linux

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

   Windows

   ```
   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 hash SHA256 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 '6BMUj6mUjjko6ZLCEdzGlWpR5sILhFJfqhW1ej3Oq1g=' | 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

```
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 user change-quorum token-sign register 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 esses 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"
      }
    ]
  }
}
```

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 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 CloudHSM CLI:

   Linux

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

   Windows

   ```
   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 sinal simbólico de quórum set-quorum-value para definir o valor mínimo do quorum. Para obter mais informações, consulte o exemplo a seguir ou use o comando help quorum token-sign set-quorum-value.

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 quorum preferido.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service user --value <2>
{
  "error_code": 0,
  "data": "Set quorum value successful"
```

```
}
```

No exemplo anterior, o serviço identifica o serviço HSM cujo valor mínimo de quorum você está configurando. O comando sinal simbólico de quórum list-quorum-values lista os tipos, nomes e descrições de serviço do HSM que estão incluídos no serviço.

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.

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

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 Usar quórum (M de N).

Usar a autenticação de quórum para administradores

Um administrador no HSM pode configurar a autenticação de quorum para as seguintes operações no cluster: AWS CloudHSM

- criar usuário
- excluir usuário
- alterar senha de usuário
- user change-mfa

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

- Obter um token de quorum
- Obter assinaturas dos administradores de aprovação
- Aprovar o token no AWS CloudHSM cluster e executar uma operação de gerenciamento de usuários

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

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

2. Use o comando login e faça login no cluster como um administrador.

   ```
   aws-cloudhsm>login --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",
  "approval_data": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJAlfK
+OWbNhZG1pbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABj5vbeAAAAAAAAAAAAAQADAAAAFQAAAAAAAAAW/
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAABAAMAAAAUA
+b23gAAAAAAAAA",
  "token": "0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": []
}
```

O caminho do arquivo consiste no seguinte:

- 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>",
        "signature": "<APPROVER_RSA2048_BIT_SIGNATURE>"
  }
  ```

  Cada assinatura é criada a partir 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.

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.

Example 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 '0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=' | 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": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJAlfK
+OWbNhZG1pbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABj5vbeAAAAAAAAAAAAQADAAAAFQAAAAAAAAAW
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAABAAMAA
+b23gAAAAAAAAAA",
  "token": "0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": [
    {
      "username": "admin2",
      "signature": "O6qx7/mUaVkYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+OsDtvYujhEU4aOFVLcrUFmyB/CX9OQmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1nOXwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWWdtDgfT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
```

```
ssktwyruGFLpXs1n0tJOEglGhx2qbYTs+omKWZdORl5WIWEXW3IXw/
Dg5vVObrNpvG0eZKO8nSMc27+cyPySc+ZbNw=="
    },
    {
      "username": "admin3",
      "signature": "O6qx7/mUaVkYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+OsDtvYujhEU4aOFVLcrUFmyB/CX9OQmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1nOXwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWWdtDgfT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyruGFLpXs1n0tJOEglGhx2qbYTs+omKWZdORl5WIWEXW3IXw/
Dg5vVObrNpvG0eZKO8nSMc27+cyPySc+ZbNw=="
    }
  ]
}
```

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 Uso da 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": []
  }
}
```

Altere o valor mínimo do quórum para administradores

Depois de definir o valor mínimo do quorum para que os admins possam usar a autenticação de quorum, você pode querer alterar o valor mínimo do quorum. 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 quorum for 2 (dois), pelo menos 2 (dois) COs deverão aprovar para alterar o valor mínimo de quorum.

> **Note**
>
> O valor do quórum do serviço ao usuário deve sempre ser menor que o valor do quórum do serviço de quórum. Para obter informações sobre nomes de serviços, como serviço de quórum e serviço de usuário, consulte Nomes e tipos de serviços que oferecem suporte à autenticação de quórum.

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.

Para alterar o valor mínimo do quórum para administradores

1. Use o comando a seguir para iniciar o modo interativo da CLI do CloudHSM.

   Linux

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

   Windows

   ```
   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 <admin> --role admin
   ```

3. Use o comando quorum token-sign list-quorum-values para obter o valor mínimo de quorum para todos os nomes de serviço. Para obter mais informações, consulte os exemplos abaixo.
4. Se o valor mínimo de quorum para o serviço do quorum for menor que o valor para o serviço ao usuário, use o comando quorum token-sign set-quorum-value para alterar o valor do serviço do quorum. Altere o valor do serviço do quorum para um (1) que seja igual ou superior ao valor do serviço ao usuário. Para obter mais informações, veja o exemplo a seguir.
5. Gere um token de quorum, tendo o cuidado de especificar o serviço de quorum como o serviço para o qual você pode usar o token.
6. Obtenha aprovações (assinaturas) de outros admins.
7. Aprove o token no AWS CloudHSM cluster e execute uma operação de gerenciamento de usuários. .
8. Use o comando quorum token-sign set-quorum-value para obter o valor mínimo de quorum para o serviço ao usuário.

Example Obtenha valores mínimos de quorum e altere o valor para serviço de quorum

O comando de exemplo a seguir mostra que o valor mínimo do quorum para o serviço ao usuário atualmente é dois (2).

```
aws-cloudhsm > quorum token-sign list-quorum-values{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 1
  }
}
```

Para alterar o valor mínimo do quorum para o serviço de quorum, use o comando quorum token-sign set-quorum-value, definindo um valor igual ou superior ao valor do serviço ao usuário. O exemplo a seguir define o valor mínimo do quorum para o serviço do quorum como dois (2), o mesmo valor definido para o serviço ao usuário.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value 2{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

O seguinte comando mostra que o valor mínimo do quorum é agora dois (2) para serviço ao usuário e serviço de quorum.

```
aws-cloudhsm > quorum token-sign list-quorum-values{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 2
  }
}
```

Gerenciar usuários do HSM com o CloudHSM Management Utility (CMU)

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 o CMU foi projetado para ser usado com a série de versões anteriores dos SDKs.

Tópicos

- Noções básicas sobre usuários do HSM
- Tabela de permissões de usuário do HSM
- Usar o CloudHSM Management Utility (CMU) para gerenciar usuários
- Usando o CloudHSM Management Utility (CMU) para gerenciar a autenticação de dois fatores (2FA) para responsáveis pela criptografia
- Como usar o CloudHSM Management Utility (CMU) para gerenciar a autenticação de quórum (controle de acesso M ou N)

Noções básicas sobre usuários do HSM

A maioria das operações executadas no HSM exigem as credenciais de um usuário do 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 HSMs interagindo com recursos 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)
- Usuário de criptografia (CU)
- 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 o cloudhsm-cli e executa o cluster activate comando. 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. Quando você ativa um novo cluster, o usuário muda de responsável pela pré-criptografia (PRECO) para responsável pela criptografia (CO).-->

Usuário de criptografia (CU)

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 Tabela de permissões de usuário do HSM.

Usuário de dispositivo (AU)

O usuário do equipamento (AU) pode realizar operações de clonagem e sincronização nos HSMs do seu cluster. AWS CloudHSM usa a AU para sincronizar os HSMs em um AWS CloudHSM cluster. A AU existe em todos os HSMs fornecidos por AWS CloudHSM, e tem permissões limitadas. Para obter mais informações, consulte Tabela de permissões de usuário do HSM.

AWS não pode realizar nenhuma operação em seus 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

A tabela a seguir lista as operações do HSM classificadas pelo tipo de usuário ou sessão do HSM que pode realizar a operação.

	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	 Não	 No (Não)	 No (Não)
Adicionar, remover usuários	 Sim	 Não	 No (Não)	 No (Não)

	Responsável pela criptografia (CO)	Usuário de criptografia (CU)	Usuário de dispositivo (AU)	Sessão não autenticada
Obter status[3] de sincronização	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Extrair, inserir objetos mascarados[4]	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Não
Funções de gerenciamento de chaves[5]	 No (Não)	 Sim	 Não	 No (Não)
Criptografar, descriptografar	 No (Não)	 Sim	 Não	 No (Não)
Assinar, verificar	 No (Não)	 Sim	 Não	 No (Não)
Gerar resumos e HMACs	 No (Não)	 Sim	 Não	 No (Não)

- [1] As informações básicas de cluster incluem o número de HSMs no cluster e cada endereço IP do HSM, o modelo, o número de série, o ID do dispositivo, o 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 arquivos de resumo para compreender o status de sincronização de 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 sincronizar os HSMs em um cluster.
- [4] As funções de gerenciamento de chaves incluem criar, excluir, encapsular, desencapsular e modificar os atributos das chaves.

Usar o CloudHSM Management Utility (CMU) para gerenciar usuários

Este tópico fornece step-by-step instruções sobre como gerenciar usuários do módulo de segurança de hardware (HSM) com o CloudHSM Management Utility (CMU), uma ferramenta de linha de comando que vem com o SDK do cliente. Para obter mais informações sobre usuários do CMU ou HSM, consulte CloudHSM Management Utility e Noções básicas sobre usuários do HSM.

Seções

- Noções básicas sobre gerenciamento de usuários do HSM com CMU
- Fazer o download de CloudHSM Management Utility
- Como gerenciar usuários do HSM com CMU

Noções básicas sobre gerenciamento de usuários do HSM com CMU

Para gerenciar os usuários do HSM, faça login no HSM com o nome de usuário e a senha de um responsável pela criptografia (CO). Somente os COs podem gerenciar usuários. O HSM contém um CO padrão chamado admin. Você define a senha para admin quando ativou o cluster.

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 ter 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/home.
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, na guia HSMs, escolha um dos endereços IP listados em Endereço IP ENI.

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`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

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.

Como gerenciar usuários do HSM com CMU

Esta seção inclui comandos básicos para gerenciar usuários do HSM com CMU.

Para criar usuários do HSM

Use createUser para criar novos usuários no 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Iniciar o CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 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

```
C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
```

2. Iniciar o CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 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.

Para listar todos os usuários do HSM no cluster

Use o comando listUsers para listar todos os usuários no 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Iniciar o CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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
               0               NO
         2              CO              example_officer                         NO
               0               NO
         3              CU              example_user                            NO
               0               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                                NO
               0               NO
         2              CO              example_officer                         NO
               0               NO
         3              CU              example_user                            NO
               0               NO
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                                NO
               0               NO
         2              CO              example_officer                         NO
               0               NO
         3              CU              example_user                            NO
               0               NO
```

Para obter mais informações sobre listUsers, consulte listUsers.

Para alterar as senhas de usuário do HSM

Use changePswd para alterar uma senha de usuário.

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 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Iniciar o CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 CO admin co12345
   ```

   Certifique-se de que o número de conexões das listas CMU corresponda ao número de HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use changePswd para alterar sua senha.

   ```
   aws-cloudhsm>changePswd CO 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(CO) 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

```
C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
```

2. Iniciar o CMU.

   Linux

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

Windows

```
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 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

Use deleteUser para excluir um usuário. Você deve fazer login como CO para excluir outro usuário.

> **Tip**
>
> 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Iniciar o CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 HSMs no cluster. Caso contrário, saia e comece de novo.

4. Use deleteUser para excluir um usuário.

   ```
   aws-cloudhsm>deleteUser CO example_officer
   ```

   A CMU exclui o usuário.

```
Deleting user example_officer(CO) 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.

Usando o CloudHSM Management Utility (CMU) para gerenciar a autenticação de dois fatores (2FA) para responsáveis pela criptografia

Para aumentar a segurança, você pode configurar a autenticação de dois fatores (2FA) para ajudar a proteger o cluster. Você só pode ativar o 2FA para responsáveis pela criptografia (CO).

> **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

- Noções básicas sobre 2FA para usuários de HSM
- Trabalhar com 2FA para usuários de HSM

Noções básicas sobre 2FA para usuários de HSM

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.

Autenticação de quórum e 2FA

O cluster usa a mesma chave para autenticação de quórum e autenticação de dois fatores (2FA). Isso significa que um usuário com 2FA ativado está efetivamente registrado no 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.

Trabalhar com 2FA para usuários de HSM

Esta seção descreve como trabalhar com 2FA para usuários do HSM, incluindo a criação de usuários do HSM 2FA, alternância de chaves e o login no HSM como usuários habilitados para 2FA. Para obter mais informações sobre como trabalhar com os usuários HSM, consulte ???, ???, ???, ??? e ???.

Criação de usuários de 2FA

Para habilitar a 2FA para um usuário do HSM, use uma chave que atenda aos seguintes requisitos.

Requisitos de pares de chaves 2FA

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 e 2FA”

Use o CMU e o par de chaves para criar um novo usuário de CO com 2FA habilitado.

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)?

   yCreating 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": "Kkdl ... 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 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: >>>>> Press Enter here
```

```
createUser success on server 0(10.0.1.11)
```

Gerenciar 2FA para usuários de HSM

Use a alteração de senha para alterar a senha de um usuário de 2FA, para ativar ou desativar a 2FA, ou para alternar a chave de 2FA. Cada vez que você habilita a 2FA, deve fornecer uma chave pública para logins de 2FA.

A alteração da senha 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 e 2FA”

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 e 2FA”.

3. Digite **y**.

   A CMU confirma a operação de alteração de senha.

Referência da configuração

Veja a seguir um exemplo das propriedades de 2FA no `authdata` arquivo para a solicitação gerada pela CMU e suas respostas.

```
{
    "Version": "1.0",
    "PublicKey": "-----BEGIN PUBLIC KEY----- ... -----END PUBLIC KEY-----",
    "Data": [
        {
            "HsmId": "hsm-lgavqitns2a",
```

```
            "Digest": "k5O1p3f6foQRVQH7S8Rrjcau6h3TYqsSdr16A54+qG8=",
            "Signature": "Kkdl ... 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)

Os HSMs em seu AWS CloudHSM cluster oferecem 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 responsáveis pela criptografia (COs): criar e excluir usuários do HSM e alterar a senha de um usuário do HSM diferente. Para ter mais informações, consulte Usar a autenticação de quorum para oficiais de criptografia.

Os seguintes tópicos fornecem mais informações sobre a autenticação de quorum no AWS CloudHSM.

Tópicos

Visão geral da autenticação de quorum

As etapas a seguir resumem os processos de autenticação de quorum. Para conhecer as etapas e ferramentas específicas, consulte Usar a autenticação de quorum para oficiais de criptografia.

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.

Detalhes adicionais sobre a autenticação de quorum

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.

Usar a autenticação de quorum para responsáveis pela criptografia: configuração inicial

Os tópicos a seguir descrevem as etapas que você deve concluir para configurar o HSM de forma que os responsáveis pela criptografia (COs) possam usar a autenticação de quorum. Você precisa realizar essas etapas apenas uma vez ao configurar inicialmente a autenticação de quorum para COs. Depois de concluir essas etapas, consulte Usar a autenticação de quorum para oficiais de criptografia.

Tópicos

- Pré-requisitos
- Criar e registrar uma chave para assinatura
- 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 Gerenciamento de usuários de HSM.

```
aws-cloudhsm>listUsers
Users on server 0(10.0.2.14):
Number of users found:7

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         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
Users on server 1(10.0.1.4):
Number of users found:7

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         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
```

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.

Example 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, substitua *officer1.key* pelo nome do arquivo onde 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 ter 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.

Example 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 esses 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 que todos os COs registrarem suas chaves públicas, o resultado do comando listUsers mostrará isso na coluna `MofnPubKey`, como mostrado no exemplo a seguir.

```
aws-cloudhsm>listUsers
Users on server 0(10.0.2.14):
Number of users found:7

    User Id             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
         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             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
         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
```

Definir o valor mínimo do quorum no HSM

Para usar a autenticação de quorum para COs, um CO deve fazer login no HSM e, em seguida, definir o valor mínimo de quorum, 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 de quorum, incluindo COs que não tenham registrado 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 ter 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) até 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 quorum 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	• setMVValue

Para obter o valor mínimo de quorum para um serviço, use o comando getMVValue, 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 getMVValue 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 Usar a autenticação de quorum para oficiais de criptografia.

Usar a autenticação de quorum para oficiais de criptografia

Um oficial de criptografia (CO) no 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 é configurado para autenticação de quorum, os COs não podem executar as operações de gerenciamento de usuários 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 para COs, faça isso agora. Para ter mais informações, consulte Configuração pela primeira vez.

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 ter 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 ter mais informações, consulte 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
```

Obter assinaturas dos COs de aprovação

Um CO que tenha um token de quorum deve obter o token aprovado por outros COs. Para darem sua aprovação, os outros COs 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 CO para assinar um resumo SHA-256 do token.

Example – Obter assinaturas dos COs de aprovação

Neste exemplo, o CO que possui o token (officer1) precisa de pelo menos duas aprovações. Os comandos do 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* and *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:
```

Aprovar o token assinado no HSM

Depois que um CO receber o número mínimo de aprovações (assinaturas) de outros COs, ele deverá 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 ter 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. Substitua *approval.txt* pelo nome do arquivo de aprovação de 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
```

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 Gerenciamento de usuários de 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
  LoginFailureCnt         2FA
         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
Users on server 1(10.0.1.4):
Number of users found:8

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         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
```

Alterar o valor mínimo do quorum para responsáveis pela criptografia

Depois de definir o valor mínimo do quorum para que os oficiais de criptografia (COs) possam usar a autenticação de quorum, você pode querer alterar o valor mínimo do quorum. 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 quorum for dois, pelo menos dois COs deverão aprovar para alterar o valor mínimo de quorum.

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 que você possa alterar o valor mínimo de quorum para os COs (serviço 3), talvez seja necessário alterar o valor mínimo de quorum 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	• createUser • deleteUser • 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 ter mais informações, consulte ???.
3. Use o comando getMVValue para obter o valor mínimo de quorum para o serviço 3. Para obter mais informações, veja o exemplo a seguir.
4. Use o comando getMVValue 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 setMVValue 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 comando setMVValue para alterar o valor mínimo do quorum 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_CO] on server 0 : [1]
MValue of service 4[MISC_CO] 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_CO] on server 0 : [2]
MValue of service 4[MISC_CO] on server 1 : [2]
```

Gerenciando chaves em AWS CloudHSM

Em AWS CloudHSM, use qualquer um dos itens a seguir para gerenciar chaves nos HSMs do 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.

Tópicos

Configurações de sincronização e durabilidade de teclas em AWS CloudHSM

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

Conceitos

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 dois HSMs.

Sincronização de chaves do lado do servidor

Clona periodicamente as chaves de cada HSM no cluster. Não requer gerenciamento.

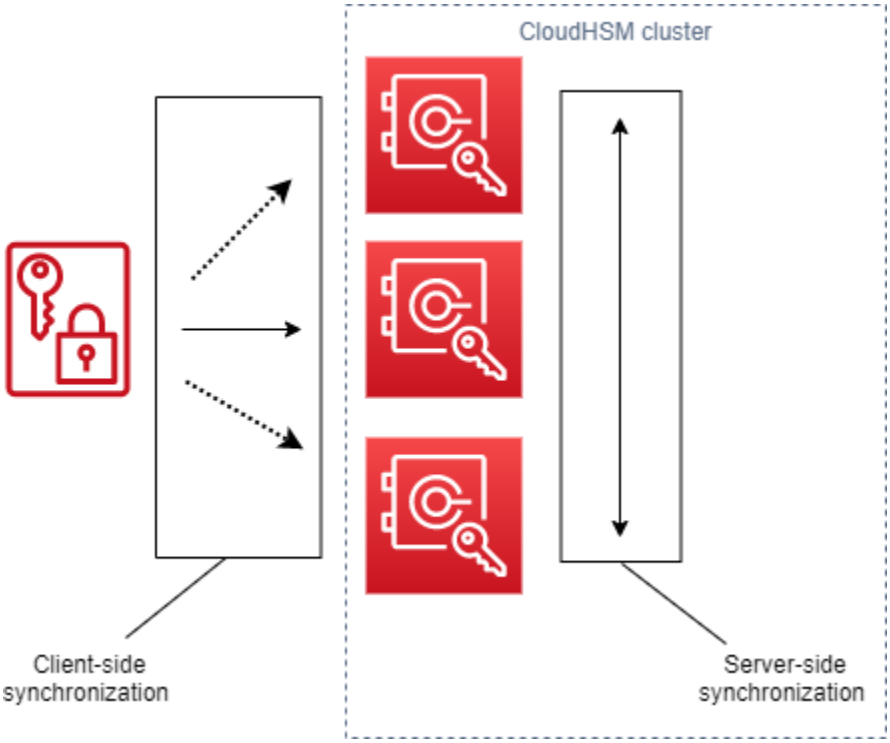
Configurações de durabilidade de chave do cliente

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 Client SDK 3, use essa configuração para especificar o número de HSMs necessários para que as operações de criação de chaves sejam bem-sucedidas.

Noções básicas sobre sincronização de teclas

AWS CloudHSM usa a sincronização de chaves para clonar chaves de token em todos os HSMs 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.

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 dois HSMs. Com a sincronização do lado do cliente e um cluster com dois HSMs, você pode atender ao desafio da durabilidade de chave em um ambiente de nuvem.

Trabalhar com configurações de durabilidade de chave do 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 de chave do lado do cliente para especificar o número de HSMs nos quais 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 dois 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 usá-las. 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.

```
"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
```

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.

```
"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.

```
"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 de HSMs nos quais 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 de 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 dois HSMs para sincronização do lado do cliente. Lembre-se de que não importa quantos HSMs você especificar, a workload 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 de HSMs nos quais 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 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 de HSMs necessários para considerar as operações de geração, importação ou desencapsulamento 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 HSMs nos quais clonar chaves. 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 obter mais informações, consulte os tópicos a seguir.

- GendSA KeyPair
- GeneCC KeyPair
- Gênero A KeyPair
- genSymKey
- importPrivateKey
- importPubKey
- imSymKey
- insertMaskedObject
- unWrapKey

Sincronização de chaves em clusters clonados

A sincronização do lado do cliente e do lado do servidor serve apenas para sincronizar chaves dentro do mesmo cluster. Se você copiar um backup de um cluster para outra região, poderá usar o comando SyncKey do cloudhsm_mgmt_util (CMU) para sincronizar chaves entre clusters. Você pode usar clusters clonados para redundância entre regiões ou para simplificar seu processo de recuperação de desastres. Para obter mais informações, consulte, syncKey.

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

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

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 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 valores permitidos para IVs, que o algoritmo de desencapsulamento 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`
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 envoltório de chaves de algoritmos chave: desembrulhar compatíveis, conforme mostrado na tabela a seguir.

- Em key_mgmt_util (KMU), use os comandos wrapKey e unWrapKey com valores m apropriados, conforme mostrado na tabela a seguir.

Algoritmo de encapsulamento de chave AES	Mecanismo PKCS #11	Método Java	Subcomando CLI do CloudHSM	Argumento do utilitário de gerenciamento de chaves (KMU)
Encapsulamento de chaves AES com preenchimento de zeros	• `CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD` (mecanismo definido pelo fornecedor)	`AESWrap/ECB/ZeroPadding`	aes-zero-pad	m = 6
Encapsulamento de chaves AES sem preenchimento	• `CKM_CLOUDHSM_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_CLOUDHSM_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

Noções básicas sobre chaves confiáveis

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 do `CKA_TRUSTED`. Além disso, administradores e responsáveis pela criptografia (COs) usam `CKA_UNWRAP_TEMPLATE` e atributos relacionados para especificar quais ações as chaves de dados podem realizar quando são desagrupadas 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 de chave confiáveis

Os atributos a seguir permitem marcar uma chave como confiável, especificar que uma chave de dados só pode ser agrupada e desagrupada com uma chave confiável e controlar o que uma chave de dados pode fazer depois de desagrupada:

- `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

Para usar uma chave confiável para agrupar uma chave de dados, 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 chave generate-asymmetric-pair .

Para obter mais informações sobre pares de chaves simétricas, consulte key generate-symmetric .

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 login para fazer login como usuário de criptografia (CU).
2. Use o comando atributo do conjunto de chaves 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. Como marcar uma chave como confiável com 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 CMU.

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 desagrupar uma chave de dados com uma chave confiável

Para desagrupar uma chave de dados, você precisa de uma chave confiável `CKA_UNWRAP` que tenha sido 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.

Gerenciando 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.

Usando a CLI do CloudHSM para gerar chaves

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
- Gerar chaves assimétricas
- Tópicos relacionados

Gerar chaves simétricas

Use os comandos listados em key generate-symmetric para gerar chaves simétricas. 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, `token=true`)

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulteAtributos de chave da CLI do CloudHSM.

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 empacotamento 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 chaves persistentes/token. 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, `token=true`)

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulteAtributos de chave da CLI do CloudHSM.

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 empacotamento 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 chaves persistentes/token. Usar <SESSION> muda isso, garantindo que uma chave gerada com esse argumento seja uma sessão/efêmera

Obrigatório: não

Gerar chaves assimétricas

Use os comandos listados em chave generate-asymmetric-pair para gerar pares de chaves assimétricas.

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 \
    --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, `token=true`).

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulteAtributos de chave da CLI do CloudHSM.

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 empacotamento 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 chaves persistentes/token. 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 `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem um limite de 126 caracteres.

Obrigatório: Sim

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário. O tamanho máximo permitido `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem um limite de 126 caracteres.

Obrigatório: Sim

<CURVE>

Especifica o identificador da curva elíptica.

Valores válidos:

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1

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, `token=true`).

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulteAtributos de chave da CLI do CloudHSM.

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, `token=true`).

Para obter uma lista dos AWS CloudHSM principais atributos compatíveis, consulteAtributos de chave da CLI do CloudHSM.

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 empacotamento 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

Tópicos relacionados

- Atributos de chave da CLI do CloudHSM
- chave generate-asymmetric-pair
- key generate-symmetric

Usando a CLI do CloudHSM para excluir chaves

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": "0x0000000000540011",
        "key-info": {
          "key-owners": [
            {
```

```
            "username": "my_crypto_user",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 CLI do CloudHSM
- excluir chave

Usar a CLI do CloudHSM para compartilhar e cancelar o compartilhamento de chaves

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 comandos key share e key unshare para compartilhar e cancelar o compartilhamento da chave com outros CUs. 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

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 ekey 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": "0x00000000001c0686",
        "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"
          },
        ],
        "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": "0x00000000001c0686",
        "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"
            }
          ],
          "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": "0x00000000001c0686",
        "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"
            },
          ],
          "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 CLI do CloudHSM
- compartilhamento de chaves
- descompartilhar chave
- Usando a CLI do CloudHSM para filtrar chaves

Usando a CLI do CloudHSM para filtrar chaves

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 e/ou filtrar chaves com a CLI do CloudHSM, os comandos de chaves utilizam um mecanismo de filtragem padronizado baseado em Atributos de chave da CLI do CloudHSM. 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": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "alice",
              "key-coverage": "full"
            }
          ],
          "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": "0x0000000000540011",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
```

```
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            }
          ],
          "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": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "my_crypto_user",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
          ],
          "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": "0x0000000000540011",
        "key-info": {
          "key-owners": [
            {
              "username": "my_crypto_user",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            }
          ],
          "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.

Example 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 CLI do CloudHSM

Como marcar uma chave como confiável com 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": "0x0000000000200333",
        "attributes": {
          "label": "test_aes_trusted"
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Usando o comando logout, faça logout como um usuário de criptografia (CU).
4. Usando o login 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"
  }
}
```

Gerenciando chaves com o KMU e o CMU

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 as chaves nos HSMs do seu AWS CloudHSM cluster usando a ferramenta de linha de comando key_mgmt_util. Antes de gerenciar as chaves, você deve iniciar o AWS CloudHSM cliente, iniciar key_mgmt_util e fazer login nos 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.

Gerar chaves

Para gerar chaves no HSM, use o comando que corresponde ao tipo de chave que você deseja gerar.

Tópicos

- Gerar chaves simétricas
- Gerar pares de chaves RSA
- Gerar pares de chaves ECC (protocolo de criptografia de curva elíptica)

Gerar chaves simétricas

Use o genSymKeycomando para gerar AES e outros tipos de chaves simétricas. 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
```

Gerar pares de chaves RSA

Para gerar um par de chaves RSA, use o comando KeyPairGenRSA. 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
```

Gerar pares de chaves ECC (protocolo de criptografia de curva elíptica)

Para gerar um key pair ECC, use o comando KeyPairGeneCC. 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

Para importar chaves secretas, isto é, chaves simétricas e chaves privadas assimétricas para o HSM, é necessário criar primeiro uma chave de encapsulamento no HSM. Importe chaves públicas diretamente sem uma chave de encapsulamento.

Tópicos

- Importar chaves secretas
- Importar chaves públicas

Importar chaves secretas

Conclua as seguintes etapas para importar uma chave secreta. 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 unWrapKeycomando.

Para importar uma chave secreta

1. Use o genSymKeycomando 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 imSymKeycomando. 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 importPrivateKeycomando. 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
```

Importar chaves públicas

Use o importPubKeycomando 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
```

Exportação de chaves

Para exportar chaves secretas, ou seja, chaves simétricas e chaves privadas assimétricas do HSM, 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 exSymKeycomando 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

- Exportar chaves secretas
- Exportar chaves públicas

Exportar chaves secretas

Conclua as seguintes etapas para exportar uma chave secreta.

Para exportar uma chave secreta

1. Use o genSymKeycomando 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 exSymKeycomando. 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

Use o comando exportPubKey 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

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 cancelar o compartilhamento de chaves

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 genSymKeyou GenRSA KeyPair. 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, ativar a end-to-end criptografia e fazer login nos 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 comando shareKey para compartilhar ou descompartilhar uma chave, especificando o identificador da chave e os IDs do usuário ou dos usuários. Para compartilhar ou descompartilhar com mais de um usuário, especifique uma lista de IDs de usuários separados por vírgula. 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 CMU

O conteúdo desta seção fornece instruções sobre como usar o CMU para marcar uma chave como confiável.

1. Usando o comando LoginHSM, faça login do como responsável pela criptografia (CO).
2. Use o comando setAttribute com `OBJ_ATTR_TRUSTED` (valor `134`) definido como true (1).

   ```
   setAttribute <Key Handle> 134 1
   ```

Gerenciando clusters clonados

Use o CloudHSM Management Utility (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

```
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 todos os HSMs e determine qual dos IPs da ENI retornados 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/home.
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, na guia HSMs, escolha um dos endereços IP listados em Endereço IP ENI.

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`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
```

```
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

Tópicos relacionados da

- syncUser
- syncKey
- Copiar um backup entre regiões

AWS CloudHSM ferramentas de linha de comando

Este tópico descreve as ferramentas de linha de comando disponíveis para gerenciamento e uso do AWS CloudHSM.

Tópicos

Compreendendo as 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 de HSM em seus 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 clusters

Comandos CloudHSMv2 AWS CLI e cmdlets PowerShell HSM2 no módulo AWSPowerShell

- Essas ferramentas obtêm, criam, excluem e marcam AWS CloudHSM clusters e HSMs:
- Para usar os comandos nos comandos do CloudHSMv2 na CLI, você precisa instalar e configurar. AWS CLI
- Os PowerShell cmdlets HSM2 no AWSPowerShell módulo estão disponíveis em um módulo Windows e em um PowerShell módulo Core multiplataforma. PowerShell

Como gerenciar usuários do HSM.

CLI do CloudHSM

- Use a CLI do CloudHSM para criar usuários, excluir usuários, listar usuários, alterar senhas de usuários e atualizar a autenticação multifatorial do usuário (MFA). 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 os 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 SDKs anteriores

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 .

Configure a ferramenta

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 os HSMs no cluster tiverem sido alterados.

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 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 a migração do SDK do cliente 3 para o SDK do cliente 5, consulte. Migração do Client SDK 3 para o Client SDK 5

Tópicos

- Ferramenta de configuração do Client SDK 5
- Ferramenta de configuração do Client SDK 3

Ferramenta de configuração do Client SDK 5

Use a ferramenta de configuração 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.

Sintaxe

PKCS #11

```
configure-pkcs11[ .exe ]
             -a <ENI IP address>
             [--hsm-ca-cert <customerCA certificate file path>]
             [--cluster-id <cluster ID>]
             [--endpoint <endpoint>]
             [--region <region>]
             [--server-client-cert-file <client certificate file path>]
             [--server-client-key-file <client key file path>]
             [--log-level <error | warn | info | debug | trace>]
                  Default is <info>
```

```
            [--log-rotation <daily | weekly>]
                 Default is <daily>
            [--log-file <file name with path>]
                 Default is </opt/cloudhsm/run/cloudhsm-pkcs11.log>
                 Default for Windows is <C:\\Program Files\\Amazon\\CloudHSM\
\cloudhsm-pkcs11.log>
            [--log-type <file | term>]
                 Default is <file>
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
            [--enable-validate-key-at-init]
                 This is the default for PKCS #11
```

OpenSSL

```
configure-dyn[ .exe ]
            -a <ENI IP address>
            [--hsm-ca-cert <customerCA certificate file path>]
            [--cluster-id <cluster ID>]
            [--endpoint <endpoint>]
            [--region <region>]
            [--server-client-cert-file <client certificate file path>]
            [--server-client-key-file <client key file path>]
            [--log-level <error | warn | info | debug | trace>]
                 Default is <error>
            [--log-type <file | term>]
                 Default is <term>
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
                 This is the default for OpenSSL
            [--enable-validate-key-at-init]
```

JCE

```
configure-jce[ .exe ]
            -a <ENI IP address>
            [--hsm-ca-cert <customerCA certificate file path>]
```

```
            [--cluster-id <cluster ID>]
            [--endpoint <endpoint>]
            [--region <region>]
            [--server-client-cert-file <client certificate file path>]
            [--server-client-key-file <client key file path>]
            [--log-level <error | warn | info | debug | trace>]
                 Default is <info>
            [--log-rotation <daily | weekly>]
                 Default is <daily>
            [--log-file <file name with path>]
                 Default is </opt/cloudhsm/run/cloudhsm-jce.log>
                 Default for Windows is <C:\\Program Files\\Amazon\\CloudHSM\
\cloudhsm-jce.log>
            [--log-type <file | term>]
                 Default is <file>
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
                 This is the default for JCE
            [--enable-validate-key-at-init]
```

CloudHSM CLI

```
configure-cli[ .exe ]
            -a <ENI IP address>
            [--hsm-ca-cert <customerCA certificate file path>]
            [--cluster-id <cluster ID>]
            [--endpoint <endpoint>]
            [--region <region>]
            [--server-client-cert-file <client certificate file path>]
            [--server-client-key-file <client key file path>]
            [--log-level <error | warn | info | debug | trace>]
                 Default is <info>
            [--log-rotation <daily | weekly>]
                 Default is <daily>
            [--log-file <file name with path>]
                 Default for Linux is </opt/cloudhsm/run/cloudhsm-cli.log>
                 Default for Windows is <C:\\Program Files\\Amazon\\CloudHSM\
\cloudhsm-cli.log>
            [--log-type <file | term>]
                 Default setting is <file>
```

```
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
                 This is the default for CloudHSM CLI
            [--enable-validate-key-at-init]
```

Configurações avançadas

Para obter uma lista de configurações avançadas específicas da ferramenta de configuração do Client SDK 5, consulte Configurações avançadas para a ferramenta de configuração do Client SDK 5.

> **Important**
>
> Depois de fazer qualquer alteração na configuração, você precisará reiniciar o aplicativo para que as alterações entrem em vigor.

Exemplos

Esses exemplos mostram como usar a ferramenta de configuração para o 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 parâmetro `-a`, você deve ter o endereço IP de um dos HSMs no 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.

```
"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>
```

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.

```
"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 IP 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.

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

> **Note**
>
> você pode usar o parâmetro `--cluster-id` no lugar de `-a <HSM_IP_ADDRESSES>`. Para ver os requisitos de uso de `--cluster-id`, consulte 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.

```
"C:\Program Files\Amazon\CloudHSM\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
```

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.

```
"C:\Program Files\Amazon\CloudHSM\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.

```
"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 em seu cluster com uma região e um 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 em seu cluster com uma região e um endpoint personalizados.

```
C:\Program Files\Amazon\CloudHSM\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 em seu cluster com uma região e um endpoint personalizados.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --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 em seu cluster com uma região e um 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 em seu cluster com uma região e um endpoint personalizados.

```
"C:\Program Files\Amazon\CloudHSM\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 em seu cluster com uma região e um 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 em seu cluster com uma região e um endpoint personalizados.

```
"C:\Program Files\Amazon\CloudHSM\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 TLS cliente-servidor

Example

Este exemplo mostra como usar os `--server-client-key-file` parâmetros `server-client-cert-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 TLS cliente-servidor com o Client SDK 5 no Linux

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` e. `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

OpenSSL Dynamic Engine

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

JCE provider

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-jce \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` e. `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
```

```
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

CloudHSM CLI

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-cli \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` e. `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

Para obter mais informações sobre os parâmetros `server-client-cert-file` e `--server-client-key-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.

```
"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
```

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.

```
"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.

```
"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 name> --log-file file --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
  ```

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> --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> --
  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-rotation`, `log-file`, `log-level` 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.

```
"C:\Program Files\Amazon\CloudHSM\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>
```

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.

```
"C:\Program Files\Amazon\CloudHSM\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.

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

Para obter mais informações sobre o parâmetro `--hsm-ca-cert`, consulte the section called “Parâmetros”.

Parâmetros

-a ***<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

--cluster-id ***<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

--endpoint ***<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

--region ***<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

-- server-client-cert-file <client certificate file path>

Caminho para o certificado de cliente usado para autenticação mútua entre cliente e servidor TLS.

Use essa opção somente se não quiser usar a chave padrão e o certificado SSL/TLS que incluímos no Client SDK 5. Você deve definir essa opção em combinação com `--server-client-key-file`.

Obrigatório: Não

-- server-client-key-file <client key file path>

Caminho para a chave do cliente usada para autenticação mútua entre cliente e servidor TLS.

Use essa opção somente se não quiser usar a chave padrão e o certificado SSL/TLS que incluímos no Client SDK 5. Você deve definir essa opção em combinação com `--server-client-cert-file`.

Obrigatório: Não

--log-level ***<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

--log-rotation ***<daily | weekly>***

Especifica a frequência com que o sistema gira os logs. Para obter mais informações, consulte Atualização do Client SDK 5.

Obrigatório: Não

--log-file ***<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

--log-type ***<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

-v, --version

Exibe a versão do .

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 dois 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 subseqüente 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

Tópicos relacionados da

- DescribeClusters Operação de API
- describe-clusters no &CLI;
- Get-HSM2Cluster PowerShell cmdlet
- Bootstrap do Client SDK 5
- AWS CloudHSM Endpoints da VPC
- Gerenciando as configurações de durabilidade da chave Client SDK 5
- Registro em log do Client SDK 5

Configurações avançadas para a ferramenta de configuração do Client SDK 5

A ferramenta de configuração do Client SDK 5 inclui configurações avançadas que não fazem parte dos atributos gerais que a maioria dos clientes utiliza. Configurações avançadas fornecem recursos adicionais.

- Configurações avançadas para PKCS #11
  - Conectando-se a vários slots com PKCS#11
  - Repetir comandos para PKCS #11
- Configurações avançadas para JCE
  - Conectando-se a vários clusters com o provedor JCE
  - Repetir comandos para o JCE
  - Extração de chaves usando JCE
- Configurações avançadas para OpenSSL
  - Repetir comandos para OpenSSL
- Configurações avançadas para interface de linha de AWS CloudHSM comando (CLI)
  - Conectando-se a vários clusters com o CloudHSM CLI

Ferramenta de configuração do Client SDK 3

Use a ferramenta de configuração do Client SDK 3 para fazer o bootstrap do daemon do cliente e configurar o CloudHSM Management Utility.

Sintaxe

```
configure -h | --help
          -a <ENI IP address>
          -m [-i <daemon_id>]
          --ssl --pkey <private key file> --cert <certificate file>
          --cmu <ENI IP address>
```

Exemplos

Esses exemplos mostram como usar a ferramenta configure.

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 parâmetro `-a`, você deve ter o endereço IP de um dos HSMs no 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/home.
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, na guia HSMs, escolha um dos endereços IP listados em Endereço IP ENI.

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`.

```
$  aws cloudhsmv2 describe-clusters
```

```
{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

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

```
C:\Program Files\Amazon\CloudHSM\bin\ 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 os HSMs do cluster no arquivo `cluster.info`.

   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

  ```
  C:\Program Files\Amazon\CloudHSM\bin\ 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

  ```
  C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
  ```

Parâmetros

-h | --help

Exibe a sintaxe do comando.

Obrigatório: Sim

-a ***<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 dos HSMs em seu cluster, use a DescribeClustersoperação, 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 os 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, em seguida, inicia o AWS CloudHSM cliente, o daemon do cliente consulta o cluster e atualiza os `cluster.info` arquivos com os endereços IP corretos do HSM para todos os HSMs no cluster. A execução do comando `-m`configure 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.

--cmu ***<ENI endereço IP>***

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

Tópicos relacionados

- Configurar a key_mgmt_util

Interface de linha de comando (CLI) do CloudHSM

A CLI do CloudHSM ajuda administradores a gerenciar usuários e usuários de criptomoedas a gerenciar chaves em seu cluster. Ele inclui ferramentas que podem ser usadas para criar, excluir e listar usuários, alterar senhas de usuários e atualizar a autenticação multifatorial do usuário (MFA). Também inclui comandos que geram, excluem, importam e exportam chaves, obtêm e definem atributos, localizam chaves e realizam operações criptográficas.

Para ver uma lista definida de usuários da CLI do CloudHSM, consulte. Gerenciando chaves com o CloudHSM CLI Para obter uma lista definida dos principais atributos da CLI do CloudHSM, consulte. Atributos de chave da CLI do CloudHSM Para obter informações sobre como usar a CLI do CloudHSM para gerenciar chaves, consulte. Gerenciando chaves com a CLI do CloudHSM

Para começar rapidamente, consulte Conceitos básicos com a Interface da Linha de Comando da CloudHSM (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

Plataformas compatíveis com a Interface da linha de comando (CLI) do CloudHSM

Suporte a Linux

Plataformas compatíveis	Arquitetura X86_64	Arquitetura ARM
Amazon Linux 2	Sim	Sim
Amazon Linux 2023	Sim	Sim
CentOS 7 (7.8+)	Sim	Não
Red Hat Enterprise Linux 7 (7.8+)	Sim	Não
Red Hat Enterprise Linux 8 (8.3+)	Sim	Não
Red Hat Enterprise Linux 9 (9.2+)	Sim	Sim
Ubuntu 20.04 LTS	Sim	Não
Ubuntu 22.04 LTS	Sim	Sim

Nota: 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

Conceitos básicos com a Interface da Linha de Comando da CloudHSM (CLI)

A interface de linha de comando (CLI) do CloudHSM permite que você gerencie 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 CLI do CloudHSM ao cluster.

Instale a CLI do CloudHSM

Use os comandos a seguir para fazer download e instalar a CLI do CloudHSM.

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 arquitetura ARM64:

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

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

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 arquitetura ARM64:

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

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

CentOS 7 (7.8+)

CentOS 7 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
```

RHEL 7 (7.8+)

RHEL 7 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
```

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 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 na arquitetura ARM64:

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

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

Ubuntu 20.04 LTS

Ubuntu 20.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-cli_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u20.04_amd64.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 arquitetura ARM64:

```
$ 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 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 msiexec.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 msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /
quiet /norestart /log C:\client-install.txt' -Wait
```

Use o comando a seguir para configurar a CLI do CloudHSM.

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 IP 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.

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

Uso da CLI do CloudHSM

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

   Linux

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

   Windows

   ```
   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**.

   É possível criar CUs porque, em uma etapa anterior, você se conectou 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 interativo e único

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

```
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.

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>
```

Por exemplo, para obter uma lista de usuários nos HSMs, insira 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

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 CLI do CloudHSM

Este tópico descreve como usar a CLI do CloudHSM para definir atributos de chave. Um atributo de chave na CLI do CloudHSM 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 chave.

Atributos compatíveis

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. Uma célula vazia na coluna Valor indica que não há nenhum valor padrão específico atribuído ao atributo.

Atributo da CLI do CloudHSM	Valor	Modificável com a key set-attribute	Configurável na criação da chave
`always-sensitive`	O valor é `True` se `sensitive` sempre esteve definido como `True` e nunca foi alterado.	Não	Não
`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` e `secp521r1`	Não	Configurável com RSA, não configurável com EC
`decrypt`	Padrão: `False`	Sim	Sim
`derive`	Padrão: `False`	Sim	Sim

Atributo da CLI do CloudHSM	Valor	Modificável com a key set-attribute	Configurável na criação da chave
`destroyable`	Padrão: `True`	Sim	Sim
`ec-point`	Para chaves EC, codificação DER do valor “Q” do ponto de eC ANSI X9.62 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	Não	Sim
`key-length-bytes`	Necessário para gerar uma chave AES. Valores válidos: 16, 24 e 32 bytes.	Não	Não
`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`	Não	Não

Atributo da CLI do CloudHSM	Valor	Modificável com a key set-attribute	Configurável na criação da chave
`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
`never-extractable`	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
`private`	Padrão: `True`	Não	Sim
`public-exponent`	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

Atributo da CLI do CloudHSM	Valor	Modificável com a key set-attribute	Configurável na criação da chave
`sensitive`	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.
`sign`	Padrão: • O valor é `True` para chaves AES. • O valor é `False` para chaves RSA e EC.	Sim	Sim
`token`	Padrão: `False`	Não	Sim
`trusted`	Padrão: `False`	Sim	Não
`unwrap`	Padrão: `False`	Sim	Sim
`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 CLI do CloudHSM	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
`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

Detalhes adicionais

Valor de verificação

O valor de verificação 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

- chave
- Referência para comandos da CLI do CloudHSM

Migrar do SDK 3 do cliente (CMU e KMU) para o SDK do cliente 5 (CloudHSM CLI)

Use este tópico para migrar fluxos de trabalho que usam as ferramentas de linha de comando do Client SDK 3, o CloudHSM Management Utility (CMU) e o Key Management Utility (KMU), para, em vez disso, usar a ferramenta de linha de comando do Client SDK 5, o CloudHSM CLI.

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 continua a ter novos recursos e suporte de plataforma adicionados a ele. Este tópico fornece detalhes específicos sobre a migração do SDK do cliente 3 para o SDK do cliente 5 para ferramentas de linha de comando.

O Client SDK 3 inclui duas ferramentas de linha de comando separadas: a 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. Interface de linha de comando (CLI) do CloudHSM As operações de gerenciamento de usuários podem ser encontradas nos subcomandos usuário e. quorum 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 de comandos.

> **Note**
>
> Se no Client SDK 3 você confiou em syncKey uma syncUser funcionalidade para sincronização entre clusters, continue usando a CMU. Atualmente, a CLI do CloudHSM no Client SDK 5 não oferece suporte a essa funcionalidade.

Para obter instruções sobre como migrar para o Client SDK 5, consulte. Migração do Client SDK 3 para o Client SDK 5 Para obter os benefícios da migração, consulteBenefícios do Client SDK 5.

Configurações avançadas para CLI

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.

- Conectando-se a vários clusters

Conectando-se a vários clusters com o CloudHSM CLI

Com o 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.

Use as instruções neste tópico para usar a CLI do CloudHSM e use a funcionalidade de vários clusters para se conectar a vários clusters.

Tópicos

- Pré-requisitos de vários clusters
- Configure a CLI do CloudHSM para funcionalidade de vários clusters
- configure-cli add-cluster
- configure-cli remove-cluster
- Usando vários clusters

Pré-requisitos de vários clusters

- 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 clusters, você já deve ter baixado e instalado a CLI do CloudHSM. Se ainda não tiver feito isso, consulte as instruções em ???.

- Você não poderá acessar um cluster configurado com, `./configure-cli[.exe] -a` pois ele não estará associado a um`cluster-id`. Você pode reconfigurá-lo seguindo `config-cli add-cluster` as instruções descritas neste guia.

Configure a CLI do CloudHSM para funcionalidade de vários clusters

Para configurar sua CLI do CloudHSM para a funcionalidade de vários clusters, siga estas etapas:

1. Identifique os clusters aos quais você deseja se conectar.
2. Adicione esses clusters à configuração da CLI do CloudHSM usando o subcomando configure-cli, conforme descrito abaixo. `add-cluster`
3. Reinicie todos os processos da CLI do CloudHSM para que a nova configuração entre em vigor.

configure-cli add-cluster

Ao se conectar a vários clusters, use o `configure-cli add-cluster` comando 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>]
        [--server-client-cert-file <CLIENT CERTIFICATE FILE>]
        [--server-client-key-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 de`cluster-1234567`) para a sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli add-cluster --cluster-id cluster-1234567
```

Windows

```
C:\Program Files\Amazon\CloudHSM\> .\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

```
C:\Program Files\Amazon\CloudHSM\> .\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

--cluster-id *<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

--endpoint *<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 do arquivo para o certificado CA do HSM.

Obrigatório: Não

--region *<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

-- server-client-cert-file <Client Certificate Filepath>

Caminho para o certificado de cliente usado para autenticação mútua entre cliente e servidor TLS.

Use essa opção somente se não quiser usar a chave padrão e o certificado SSL/TLS que incluímos no Client SDK 5. Você deve definir essa opção em combinação com `--server-client-key-file`.

Obrigatório: Não

-- server-client-key-file <Client Key Filepath>

Caminho para a chave do cliente usada para autenticação mútua entre cliente e servidor TLS.

Use essa opção somente se não quiser usar a chave padrão e o certificado SSL/TLS que incluímos no Client SDK 5. Você deve definir essa opção em combinação com `--server-client-cert-file`.

Obrigatório: Não

configure-cli remove-cluster

Ao se conectar a vários clusters com a CLI do CloudHSM, `configure-cli remove-cluster` use o comando 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

```
C:\Program Files\Amazon\CloudHSM\> .\configure-cli.exe remove-cluster --cluster-
id cluster-1234567
```

Para obter mais informações sobre o parâmetro `--cluster-id`, consulte the section called “Parâmetros”.

Parâmetro

--cluster-id ***<Cluster ID>***

O ID do cluster a ser removido da configuração.

Obrigatório: Sim

Usando vários clusters

Depois de configurar vários clusters com a CLI do CloudHSM, use `cloudhsm-cli` o comando para interagir com eles.

Exemplos

Definindo um padrão **`cluster-id`** ao usar o modo interativo

Example

Use o ??? junto com o `cluster-id` parâmetro para definir um cluster padrão (com o ID `decluster-1234567`) da sua configuração.

Linux

```
$ cloudhsm-cli interactive --cluster-id cluster-1234567
```

Windows

```
C:\Program Files\Amazon\CloudHSM\> .\cloudhsm-cli.exe interactive --cluster-
id cluster-1234567
```

Configurando o **`cluster-id`** ao executar um único comando

Example

Use o `cluster-id` parâmetro para definir o cluster (com o ID `decluster-1234567`) ??? do qual obter.

Linux

```
$ cloudhsm-cli cluster hsm-info --cluster-id cluster-1234567
```

Windows

```
C:\Program Files\Amazon\CloudHSM\> .\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 Conceitos básicos com a Interface da Linha de Comando da CloudHSM (CLI).

Para executar a maioria dos comandos da CLI do CloudHSM, você deve iniciar a CLI do CloudHSM e fazer login no HSM. Se adicionar ou excluir HSMs, atualize os arquivos de configuração da CLI do CloudHSM. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os HSMs no cluster.

Os seguintes tópicos descrevem comandos na CLI do CloudHSM:

Command	Descrição	Tipo de usuário
cluster 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
cluster hsm-info	Liste os HSMs em seu cluster.	Tudo [1], incluindo usuários não autenticados. O login não é necessário.
sinal criptográfico ecdsa	Gera uma assinatura usando uma chave privada EC e o mecanismo de assinatura ECDSA.	Usuários de criptografia (CU)
sinal criptográfico rsa-pkcs	Gera uma assinatura usando uma chave privada RSA e o mecanismo de assinatura RSA-PKCS.	CU
sinal criptográfico rsa-pkcs-pss	Gera uma assinatura usando uma chave privada RSA e o mecanismo de assinatura RSA-PKCS-PSS.	CU
verificação criptográfica ecdsa	Confirma que um arquivo foi assinado no HSM por uma determinada chave pública. Verifica se a assinatur	CU

Command	Descrição	Tipo de usuário
	a foi gerada usando o mecanismo de assinatura ECDSA. Compara um arquivo assinado com um arquivo de origem e determina se os dois estão relacionados criptograficamente com base em uma determinada chave pública ecdsa e mecanismo de assinatura.	
verificação criptográfica 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 relacionados criptogra ficamente com base em uma determinada chave pública rsa e mecanismo de assinatura.	CU

Command	Descrição	Tipo de usuário
verificação criptográfica rsa-pkcs-pss	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-PSS. Compara um arquivo assinado com um arquivo de origem e determina se os dois estão relacionados criptogra ficamente com base em uma determinada chave pública rsa e mecanismo de assinatura.	CU
excluir chave	Exclui uma chave do seu AWS CloudHSM cluster.	CU
key generate-file	Gera um arquivo de chave no seu AWS CloudHSM cluster.	CU
chave generate-asymmetric-pair rsa	Gera um par de chaves RSA assimétrico em seu cluster. AWS CloudHSM	CU
chave, generate-asymmetric-pair etc.	Gera um par de chaves assimétrica de curva elíptica (EC) em seu cluster. AWS CloudHSM	CU
key generate-symmetric aes	Gera uma chave AES simétrica em seu AWS CloudHSM cluster.	CU
key generate-symmetric generic-secret	Gera uma chave secreta genérica simétrica em seu AWS CloudHSM cluster.	CU

Command	Descrição	Tipo de usuário
pem de importação de chaves	Importa uma chave de 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.	Os CUs podem executar esse comando, os administradores podem definir o atributo confiável.
compartilhamento de chaves	Compartilha uma chave com outras CUs em seu AWS CloudHSM cluster.	CU
descompartilhar chave	Não compartilha uma chave com outras CUs em seu AWS CloudHSM cluster.	CU
chave unwrap aes-gcm	Desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o mecanismo de desempacotamento AES-GCM.	CU

Command	Descrição	Tipo de usuário
chave: desembrulhar 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
chave unwrap aes-pkcs5-pad	Desempacota uma chave de carga útil usando a chave de empacotamento AES e o mecanismo de desempaco tamento AES-PKCS5-PAD.	CU
chave: desembrulhar 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
chave: desembrulhar cloudhsm-aes-gcm	Desempacota uma chave de carga útil no cluster usando a chave de empacotam ento AES e o mecanismo de desempacotamento CLOUDHSM-AES-GCM.	CU
chave unwrap rsa-aes	Desempacota uma chave de carga usando uma chave privada RSA e o mecanismo de desempacotamento RSA-AES.	CU

Command	Descrição	Tipo de usuário
chave unwrap rsa-oaep	Desempacota uma chave de carga usando a chave privada RSA e o mecanismo de desempacotamento RSA-OAEP.	CU
chave unwrap rsa-pkcs	Desempacota uma chave de carga usando a chave privada RSA e o mecanismo de desempacotamento RSA-PKCS.	CU
envoltório de chaves aes-gcm	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de encapsulamento AES-GCM.	CU
envoltório de chaves aes-no-pad	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de empacotamento AES-NO-PAD.	CU
envoltório de chaves aes-pkcs5-pad	Encapsula uma chave de carga usando uma chave AES no HSM e o mecanismo de encapsulamento AES-PKCS5-PAD.	CU
envoltório de chaves aes-zero-pad	Encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de encapsulamento AES-ZERO-PAD.	CU

Command	Descrição	Tipo de usuário
envoltório de chaves cloudhsm-aes-gcm	Encapsula uma chave de carga usando uma chave AES no HSM e o mecanismo de encapsulamento CLOUDHSM-AES-GCM.	UCs
envoltório de chaves rsa-aes	Encapsula uma chave de carga útil usando uma chave pública RSA no HSM e o mecanismo de empacotam ento RSA-AES.	CU
envoltório de chaves rsa-oaep	Encapsula uma chave de carga usando uma chave pública RSA no HSM e o mecanismo de empacotam ento RSA-OAEP.	CU

Command	Descrição	Tipo de usuário
O key wrap rsa-pkcs comando agrupa uma chave de carga usando uma chave pública RSA no HSM e o mecanismo de agrupamento. `RSA-PKCS` O `extractable` atributo da chave de carga útil deve ser definido como. `true` Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 chave generate-asymmetric-pair comando e o `wrap` atributo definido como. `true` Tipo de usuário Os seguintes tipos de usuários podem executar este comando. • Usuários de criptografia (CUs) Requisitos • Para executar esse comando, você deve estar	Encapsula uma chave de carga usando uma chave pública RSA no HSM e o mecanismo de empacotamento RSA-PKCS.	CU

Command	Descrição	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.
quorum token-sign list-timeouts	Obtém o período de tempo limite do token em segundos para todos os tipos de token.	Administrador e usuário de criptografia
sinal simbólico de quórum set-quorum-value	Define um novo valor de quórum para um serviço autorizado por quórum.	Administrador
quorum token-sign set-timeout	Configura o período de tempo limite do token em segundos para todos os tipos de token.	Administrador

Command	Descrição	Tipo de usuário
user change-mfa	Altera a estratégia de autenticação multifator (MFA) do usuário.	Administrador, CU
alterar senha de usuário	Altera as senhas dos usuários nos 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
ista 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.

cluster

cluster é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando específico para os usuários. Atualmente, a categoria de usuário consiste nos seguintes comandos:

- cluster activate
- cluster hsm-info

cluster activate

Use o comando cluster activate na CLI do CloudHSM para ativar um novo cluster. 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 da

- criar usuário
- excluir usuário
- alterar senha de usuário

cluster hsm-info

Use o comando cluster hsm-info na CLI do CloudHSM para listar HSMs no seu cluster. 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 que você fizer podem não ser efetivas em todos os 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

```
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 os HSMs presentes em seu AWS CloudHSM cluster.

```
aws-cloudhsm > cluster hsm-info
{
  "error_code": 0,
  "data": {
    "hsms": [
      {
        "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.
- H ardware-version-major: A versão principal do hardware.
- H ardware-version-minor: A versão secundária do hardware.
- F irmware-version-major: A versão principal do firmware.
- F irmware-version-minor: A versão secundária do firmware.
- F irmware-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 dos HSMs nele. Se estiver no modo FIPS, a saída será “2 [modo FIPS com autenticação de fator único]”. Se estiver no modo não FIPS, a saída será “0 [modo não FIPS com autenticação de fator único]”.

Tópicos relacionados

- cluster activate

crypto

cryptoé uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando específico para operações criptográficas. Atualmente, essa categoria consiste nos seguintes comandos:

- sinal criptográfico
  - sinal criptográfico ecdsa
  - sinal criptográfico rsa-pkcs
  - sinal criptográfico rsa-pkcs-pss
- verificação criptográfica
  - verificação criptográfica ecdsa
  - verificação criptográfica rsa-pkcs
  - verificação criptográfica rsa-pkcs-pss

sinal criptográfico

crypto signé uma categoria principal para um grupo de comandos que, quando combinado com a categoria principal, usa uma chave privada escolhida em seu AWS CloudHSM cluster para gerar uma assinatura. crypto signtem os seguintes subcomandos:

- sinal criptográfico ecdsa
- sinal criptográfico rsa-pkcs
- sinal criptográfico rsa-pkcs-pss

Para usarcrypto sign, você deve ter uma chave privada em seu HSM. Você pode gerar uma chave privada com os seguintes comandos:

- chave, generate-asymmetric-pair etc.
- chave generate-asymmetric-pair rsa

sinal criptográfico ecdsa

O crypto sign ecdsa comando gera 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 chave, generate-asymmetric-pair etc. comando com o `sign` atributo definido como`true`.

> **Note**
>
> As assinaturas podem ser verificadas AWS CloudHSM com verificação criptográfica subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Esses exemplos mostram como usar crypto sign ecdsa para gerar uma assinatura usando o mecanismo de assinatura ECDSA e a função SHA256 hash. 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": "0x0000000000007808dd",
    "signature": "4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2wOS3Q5bzk
+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/KqoQvh4ueMAxQQVp7FQguZ2wOS3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

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 a localização 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, referência de chave = 0xabc) ou lista separada por espaço 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, consulte Atributos principais da CLI do CloudHSM.

Obrigatório: Sim

Tópicos relacionados

- sinal criptográfico

- verificação criptográfica

sinal criptográfico rsa-pkcs

O crypto sign rsa-pkcs comando gera 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 chave generate-asymmetric-pair rsa comando com o `sign` atributo `true` definido como.

> **Note**
>
> As assinaturas podem ser verificadas AWS CloudHSM com verificação criptográfica subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Esses exemplos mostram como usar 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": "0x0000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+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": "0x00000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

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 a localização 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 principais atributos da CLI do CloudHSM, consulte Atributos principais da CLI do CloudHSM.

Obrigatório: Sim

Tópicos relacionados

- sinal criptográfico
- verificação criptográfica

sinal criptográfico rsa-pkcs-pss

O crypto sign rsa-pkcs-pss comando gera uma assinatura usando uma chave privada RSA e o mecanismo de `RSA-PKCS-PSS` assinatura.

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 chave generate-asymmetric-pair rsa comando com o `sign` atributo `true` definido como.

> **Note**
>
> As assinaturas podem ser verificadas AWS CloudHSM com verificação criptográfica subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help                           Print help
```

Exemplo

Esses exemplos mostram como usar crypto sign rsa-pkcs-pss para gerar uma assinatura usando o mecanismo de RSA-PKCS-PSS assinatura e a função SHA256 hash. Esse comando usa uma chave privada no HSM.

Example Exemplo: gerar uma assinatura para dados codificados de 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": "0x00000000007008db",
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
```

```
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg=="
  }
}
```

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": "0x00000000007008db",
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

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 a localização 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 principais atributos da CLI do CloudHSM, consulte Atributos principais da CLI do CloudHSM.

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

Tópicos relacionados

- sinal criptográfico
- verificação criptográfica

Tópicos relacionados

- verificação criptográfica

verificação criptográfica

crypto verifyé uma categoria principal para um grupo de comandos que, quando combinada com a categoria principal, confirma se um arquivo foi assinado por uma determinada chave. crypto verifytem os seguintes subcomandos:

- verificação criptográfica ecdsa
- verificação criptográfica rsa-pkcs
- verificação criptográfica rsa-pkcs-pss

O crypto verify comando compara um arquivo assinado com um arquivo de origem e analisa se eles estão relacionados criptograficamente com base em uma determinada chave pública e mecanismo de assinatura.

> **Note**
>
> Os arquivos podem ser conectados AWS CloudHSM com a sinal criptográfico operação.

verificação criptográfica ecdsa

O crypto verify ecdsa comando é usado 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 determine se os dois estão relacionados criptograficamente com base em uma determinada chave pública ecdsa e mecanismo de assinatura.

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 pem de importação de chaves comando com o `verify` atributo definido como`true`.

> **Note**
>
> Você pode gerar uma assinatura na CLI sinal criptográfico do CloudHSM com subcomandos.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Esses exemplos mostram como usar crypto verify ecdsa para verificar uma assinatura que foi gerada usando o mecanismo de assinatura ECDSA 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 ecdsa --hash-function sha256 --key-filter attr.label=ec-
public --data YWJjMTIz --signature 4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2wOS3Q5bzk
+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: Prove uma relação de assinatura falsa

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 uma relação de assinatura verdadeira, o comando retorna uma mensagem de erro.

```
aws-cloudhsm > crypto verify ecdsa --hash-function sha256 --
key-filter attr.label=ec-public --data aW52YWxpZA== --signature
 +ogk7M7S3iTqFg3SndJfd91dZFr5Qo6YixJl8JwcvqqVgsVuO6o+VKvTRjz0/V05kf3JJbBLr87Q
+wLWcMAJfA==
{
  "error_code": 1,
  "data": "Signature verification failed"
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

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 a localização 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 principais atributos da CLI do CloudHSM, consulte Atributos principais da CLI do CloudHSM.

Obrigatório: Sim

<SIGNATURE>

Assinatura codificada em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho da assinatura)

<SIGNATURE_PATH>

Especifica a localização da assinatura.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho da assinatura)

Tópicos relacionados

- sinal criptográfico
- verificação criptográfica

verificação criptográfica rsa-pkcs

O crypto verify rsa-pkcs comando é usado 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 RSA-PKCS assinatura.
- Compare um arquivo assinado com um arquivo de origem e determine se os dois estão relacionados criptograficamente com base em uma determinada chave pública rsa e 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 CLI do CloudHSM com os subcomandos. sinal criptográfico

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -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
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+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: Prove uma relação de assinatura falsa

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 uma relação de assinatura verdadeira, 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
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+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 a localização 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 principais atributos da CLI do CloudHSM, consulte Atributos principais da CLI do CloudHSM.

Obrigatório: Sim

<SIGNATURE>

Assinatura codificada em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho da assinatura)

<SIGNATURE_PATH>

Especifica a localização da assinatura.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho da assinatura)

Tópicos relacionados

- sinal criptográfico

- verificação criptográfica

verificação criptográfica rsa-pkcs-pss

O crypto sign rsa-pkcs-pss comando é usado 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-PSS.
- Compare um arquivo assinado com um arquivo de origem e determine se os dois estão relacionados criptograficamente com base em uma determinada chave pública rsa e 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 `verify` atributo definido como. `true`

> **Note**
>
> Você pode gerar uma assinatura usando a CLI do CloudHSM com os subcomandos. sinal criptográfico

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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 assinatura RSA-PKCS-PSS 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-pss --key-filter attr.label=rsa-public
 --hash-function sha256 --data YWJjMTIz --salt-length 10 --mgf mgf1-sha256
```

```
 --signature H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg==
{
  "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: Prove uma relação de assinatura falsa

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 uma relação de assinatura verdadeira, 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/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg==
{
  "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 a localização 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 principais atributos da CLI do CloudHSM, consulte Atributos principais da CLI do CloudHSM.

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 a localização da assinatura.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho da assinatura)

Tópicos relacionados

- sinal criptográfico
- verificação criptográfica

chave

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 chave
- key 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
- pem de importação de chaves
- lista de chaves
- replicação de chave
- atributo do conjunto de chaves
- compartilhamento de chaves
- descompartilhar chave
- chave: desembrulhar
- envoltório de chaves

excluir chave

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, consequentemente, 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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 no qual executar essa operação.

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` 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 CLI do CloudHSM.

Obrigatório: Sim

Tópicos relacionados

- lista de chaves
- key generate-file
- descompartilhar chave
- Atributos de chave da CLI do CloudHSM
- Usando a CLI do CloudHSM para filtrar chaves

key generate-file

O key generate-file comando exporta uma chave assimétrica do HSM. Se o destino for uma chave privada, a referência à chave privada será exportada em 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 descarregamento de SSL/TLS 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 criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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` selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis com com a CLI do CloudHSM, consulte Atributos de chave da CLI do CloudHSM

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

Tópicos relacionados

- Atributos de chave da CLI do CloudHSM
- Usando a CLI do CloudHSM para filtrar chaves
- chave generate-asymmetric-pair
- key generate-symmetric

chave generate-asymmetric-pair

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

chave, generate-asymmetric-pair etc.

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 criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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]
      --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
  -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": "0x000000000012000b",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",
        "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":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f38a
        "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": "ec-private-key-example",
        "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": 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 token=true encrypt=true \
    --private-attributes token=true decrypt=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000002806eb",
      "key-info": {
        "key-owners": [
          {
```

```
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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":
"0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
        "curve": "secp224r1"
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280c82",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
```

```
          }
        ],
        "shared-users": [],
        "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"
      }
    }
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

Obrigatório: se vários clusters tiverem sido configurados.

<CURVE>

Especifica o identificador da curva elíptica.

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1

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, `token=true`)

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

Obrigatório: não

<PUBLIC_LABEL>

Especifica um rótulo definido pelo usuário para a chave pública. O tamanho máximo permitido `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem 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, `token=true`)

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

Obrigatório: não

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário. O tamanho máximo permitido `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem 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 empacotamento 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

Tópicos relacionados

- Atributos de chave da CLI do CloudHSM
- Usando a CLI do CloudHSM para filtrar chaves

chave generate-asymmetric-pair rsa

O uso do key generate-asymmetric-pair rsa comando gera 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 criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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
  -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": "0x0000000000160010",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "",
        "check-value": "0x498e1f",
        "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": 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": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-private-key-example",
        "id": "",
        "check-value": "0x498e1f",
        "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": 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 token=true encrypt=true \
--private-attributes token=true decrypt=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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": "0x0000000000280cc7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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
      }
    }
  }
}
```


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, `token=true`)

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

Obrigatório: não

<PRIVATE_LABEL>

Especifica o rótulo da chave privada definida pelo usuário. O tamanho máximo permitido `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem 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, `token=true`)

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

Obrigatório: não

<PUBLIC_LABEL>

Especifica um rótulo definido pelo usuário para a chave pública. O tamanho máximo permitido `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem 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 empacotamento 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

Tópicos relacionados

- Atributos de chave da CLI do CloudHSM
- Usando a CLI do CloudHSM para filtrar chaves

key generate-symmetric

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

key generate-symmetric aes

O key generate-symmetric aes comando gera uma chave AES simétrica em seu AWS CloudHSM cluster.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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
  -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": "0x00000000002e06bf",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "aes",
        "label": "example-aes",
        "id": "",
        "check-value": "0x9b94bd",
        "class": "secret-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": 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": "0x00000000002e06bf",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "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 no qual executar essa operação.

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, `token=true`).

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

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 `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem 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 empacotamento 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

Tópicos relacionados

- Atributos de chave da CLI do CloudHSM
- Usando a CLI do CloudHSM para filtrar chaves

key generate-symmetric generic-secret

O comando key generate-asymmetric-pair gera uma chave secreta genérica simétrica no seu cluster AWS CloudHSM.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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
  -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": "0x00000000002e08fd",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
```

```
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",
        "class": "secret-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": 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 token=true encrypt=true
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e08fd",
```

```
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "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, `token=true`).

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

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 `label` é de 127 caracteres para o SDK do cliente 5.11 e versões posteriores. O SDK do cliente 5.10 e versões anteriores tem 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 empacotamento 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

Tópicos relacionados

- Atributos de chave da CLI do CloudHSM
- Usando a CLI do CloudHSM para filtrar chaves

pem de importação de chaves

O key import pem comando em AWS CloudHSM importa uma chave de formato PEM para um HSM. Você pode usá-lo para importar chaves públicas que foram geradas fora do HSM.

> **Note**
>
> Use o key generate-file comando 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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 key import pem comando 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": "0x00000000001e08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "example-imported-key",
        "id": "0x",
        "check-value": "0x99fe93",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
```

```
        "token": false,
        "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":
 "0x8e9c172c37aa22ed1ce25f7c3a7c936dadc532201400128b044ebb4b96#··3e4930ab910df5a2896eaeb8853cfe
        "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": "0x00000000001e08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
```

```
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "example-imported-key-with-attributes",
        "id": "0x",
        "check-value": "0x99fe93",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": false,
        "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":
 "0x8e9c172c37aa22ed1ce25f7c3a7c936dadc532201400128b044ebb4b96#··3e4930ab910df5a2896eaeb8853cfe
        "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 do arquivo em que o arquivo de chave está localizado.

Obrigatório: Sim

<LABEL>

Especifica um rótulo definido pelo usuário para a chave importada. O tamanho máximo para `label` é de 126 caracteres.

Obrigatório: Sim

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave embrulhada.

Possíveis valores:

- ec-público
- rsa-public

Obrigatório: Sim

<IMPORT_KEY_ATTRIBUTES>

Especifica uma lista separada por espaços de atributos de chave a serem definidos para a chave importada na forma de `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (por exemplo,`token=true`). Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

Obrigatório: não

Tópicos relacionados

- sinal criptográfico
- verificação criptográfica

lista de chaves

O key list comando encontra todas as chaves do usuário atual presente em seu AWS CloudHSM cluster. 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.

- Usuários de criptografia (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
  -h, --help
          Print help
```

Exemplos

Os exemplos a seguir mostram as diferentes maneiras de executar o comando key list.

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": "0x00000000000003d5",
        "attributes": {
          "label": "test_label_1"
        }
      },
      {
        "key-reference": "0x0000000000000626",
        "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 - detalhadas

A saída inclui chaves que o usuário possui e compartilha, bem como todas as chaves públicas nos 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": "0x000000000012000c",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "session"
        },
        "attributes": {
          "key-type": "ec",
          "label": "ec-test-private-key",
          "id": "",
          "check-value": "0x2a737d",
          "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": false,
          "trusted": false,
          "unwrap": false,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 122,
          "ec-point":
"0x0442d53274a6c0ec1a23c165dcb9ccdd72c64e98ae1a9594bb5284e752c746280667e11f1e983493c1c605e0a80
          "curve": "secp224r1"
        }
      },
      {
        "key-reference": "0x000000000012000d",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "cluster-coverage": "session"
        },
        "attributes": {
          "key-type": "ec",
          "label": "ec-test-public-key",
          "id": "",
          "check-value": "0x2a737d",
          "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":
 "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": "0x0000000000000030",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "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": "0x0000000000000042",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "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": "0x0000000000000081",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
```

```
            }
          ],
          "shared-users": [],
          "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": "0x0000000000000089",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "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 CLI do CloudHSM, consulte Usando a CLI do CloudHSM para filtrar chaves.

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

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` selecionar chaves correspondentes à lista.

Para obter uma lista dos atributos de chave compatíveis com com a CLI do CloudHSM, consulte Atributos de chave da CLI do CloudHSM

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.

Obrigatório: não

Tópicos relacionados

- excluir chave
- key generate-file
- descompartilhar chave
- Atributos de chave da CLI do CloudHSM

- Usando a CLI do CloudHSM para filtrar chaves

replicação de chave

O key replicate comando replica uma chave de um cluster de origem para um AWS CloudHSM cluster de destino AWS CloudHSM .

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (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 Criação de clusters a partir de backups Para mais informações.
- O proprietário da chave deve existir no cluster de destino. Além disso, se a chave for compartilhada com qualquer usuário, esses usuários também deverão existir no cluster de destino.
- Para executar esse comando, você deve estar logado como uma UC 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 mais informações. Para fornecer credenciais para o cluster de destino, você precisa definir duas variáveis ambientais adicionais: DESTINATION_CLOUDHSM_PIN e DESTINATION_CLOUDHSM_ROLE:

```
$ export DESTINATION_CLOUDHSM_ROLE=crypto-user
```

```
$ 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: chave de replicação

Esse comando replica uma chave de um cluster de origem para um cluster de destino clonado.

```
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": "0x0000000000300006",
      "key-info": {
        "key-owners": [
          {
            "username": "crypto-user-1",
```

```
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave correspondente no cluster de origem.

Para obter uma lista dos atributos de chave compatíveis com com a CLI do CloudHSM, consulte Atributos de chave da CLI do CloudHSM

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

- Conectando-se a vários clusters com o CloudHSM CLI

atributo do conjunto de chaves

Use o key set-attribute comando para definir os atributos das chaves em seu AWS CloudHSM cluster. Somente o CU que criou a chave e, consequentemente, a possui pode alterar os atributos da chave.

Para obter uma lista dos principais atributos que podem ser usados na CLI do CloudHSM, consulte Atributos de chave da CLI do CloudHSM.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Somente usuários de criptografia (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
  -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": "0x0000000000006400ec",
        "key-info": {
          "key-owners": [
            {
              "username": "bob",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "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 no qual executar essa operação.

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` selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis com com a CLI do CloudHSM, consulte Atributos de chave da CLI do CloudHSM

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

Tópicos relacionados

- Usando a CLI do CloudHSM para filtrar chaves
- Atributos de chave da CLI do CloudHSM

compartilhamento de chaves

O key share comando compartilha uma chave com outras CUs em seu AWS CloudHSM cluster.

Somente a UC que criou a chave e, consequentemente, 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 criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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

-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": "0x00000000001c0686",
        "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"
            }
          ],
          "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 no qual executar essa operação.

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` selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

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 Noções básicas sobre usuários do HSM.

Obrigatório: Sim

Tópicos relacionados

- Usando a CLI do CloudHSM para filtrar chaves
- Atributos de chave da CLI do CloudHSM

descompartilhar chave

O key unshare comando cancela o compartilhamento de uma chave com outras CUs em seu AWS CloudHSM cluster.

Somente o CU que criou a chave e, consequentemente, 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 criptografia (CUs)

Requisitos

Para executar esse comando, você deve estar registrado 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

  -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": "0x00000000001c0686",
        "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"
          }
        ],
        "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": "0x00000000001c0686",
        "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"
          },
        ],
        "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
  }
}
```

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` selecionar uma chave correspondente para exclusão.

Para obter uma lista dos atributos de chave compatíveis, consulte Atributos de chave da CLI do CloudHSM.

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 Noções básicas sobre usuários do HSM.

Obrigatório: Sim

Tópicos relacionados

- Usando a CLI do CloudHSM para filtrar chaves
- Atributos de chave da CLI do CloudHSM

chave: desembrulhar

O comando key unwrap pai na CLI do CloudHSM importa uma chave privada simétrica ou assimétrica criptografada (encapsulada) de um arquivo para o HSM. Esse comando foi projetado para importar chaves criptografadas que foram agrupadas pelo envoltório de chaves comando, mas também pode ser usado para desempacotar chaves que foram agrupadas com outras ferramentas. No entanto, nessas situações, recomendamos usar as bibliotecas de software PKCS#11 ou JCE para desencapsular a chave.

- chave unwrap aes-gcm
- chave: desembrulhar aes-no-pad
- chave unwrap aes-pkcs5-pad
- chave: desembrulhar aes-zero-pad
- chave: desembrulhar cloudhsm-aes-gcm
- chave unwrap rsa-aes
- chave unwrap rsa-oaep
- chave unwrap rsa-pkcs

chave unwrap aes-gcm

O key unwrap aes-gcm comando desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o mecanismo de desempacotamento. `AES-GCM`

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap aes-gcm comando usando uma chave AES com o valor do `unwrap` atributo definido como`true`.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados 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 xvslgrtg8kHzrvekny97tLSIeokpPwV8
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001808e4",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001808e4",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
```

```
          }
        ],
        "shared-users": [],
        "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 no qual executar essa operação.

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 selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave na forma de `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para a chave encapsulada.

Obrigatório: não

<AAD>

Como valor de Dados Autenticados Adicionais (AAD) do GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Aes o comprimento da tag GCM em bits.

Obrigatório: Sim

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: `aesdes3`,`ec-private`,,`generic-secret`,`rsa-private`].

Obrigatório: Sim

<LABEL>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente 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

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave: desembrulhar aes-no-pad

O key unwrap aes-no-pad comando desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o 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 eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap aes-no-pad comando usando uma chave AES com o valor do `unwrap` atributo definido como`true`.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados em Base64

```
aws-cloudhsm > key unwrap aes-no-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data eXK3PMAOnKM9y3YX6brbhtMoC060EOH9
```

```
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08ec",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001c08ec",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 de atributos de chave na forma de selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave 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: `aesdes3`,`ec-private`,,`generic-secret`,`rsa-private`].

Obrigatório: Sim

<LABEL>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave unwrap aes-pkcs5-pad

O key unwrap aes-pkcs5-pad comando desempacota uma chave de carga usando a chave de empacotamento AES e o mecanismo de desempacotamento. `AES-PKCS5-PAD`

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap aes-pkcs5-pad comando usando uma chave AES com o valor do `unwrap` atributo definido como`true`.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados em Base64

```
aws-cloudhsm > key unwrap aes-pkcs5-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data MbuYNresfOKyGNnxKWen88nSfX+uUE/0qmGofSisicY=
```

```
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001c08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 no qual executar essa operação.

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 selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave 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: `aesdes3`,`ec-private`,,`generic-secret`,`rsa-private`].

Obrigatório: Sim

<LABEL>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave: desembrulhar aes-zero-pad

O key unwrap aes-zero-pad comando desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o mecanismo de desempacotamento. `AES-ZERO-PAD`

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap aes-zero-pad comando usando uma chave AES com o valor do `unwrap` atributo definido como`true`.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados em Base64

```
aws-cloudhsm > key unwrap aes-zero-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data L1wVlL/YeBNVAw6Mpk3owFJZXBzDLONt
```

```
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08e7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001c08e7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 de atributos de chave na forma de selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave 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: `aesdes3`,`ec-private`,,`generic-secret`,`rsa-private`].

Obrigatório: Sim

<LABEL>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave: desembrulhar cloudhsm-aes-gcm

O key unwrap cloudhsm-aes-gcm comando desempacota uma chave de carga útil no cluster usando a chave de empacotamento AES e o 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 eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap cloudhsm-aes-gcm comando usando uma chave AES com o valor do `unwrap` atributo definido como`true`.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados 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
 6Rn8nkjEriDYlnP3P8nPkYQ8hplOEJ899zsrF+aTB0i/fIlZ
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001408e8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001408e8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 no qual executar essa operação.

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 selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave na forma de `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` para a chave encapsulada.

Obrigatório: não

<AAD>

Como valor de Dados Autenticados Adicionais (AAD) do GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Aes o comprimento da tag GCM em bits.

Obrigatório: Sim

<KEY_TYPE_CLASS>

Tipo de chave e classe da chave encapsulada [valores possíveis: `aesdes3`,`ec-private`,,`generic-secret`,`rsa-private`].

Obrigatório: Sim

<LABEL>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave unwrap rsa-aes

O key unwrap rsa-aes comando desempacota uma chave de carga usando uma chave privada RSA e o mecanismo de desempacotamento. `RSA-AES`

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que eles não foram gerados localmente, seu `local` atributo é definido como`false`.

Para usar okey 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`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplo

Esses exemplos mostram como usar o key unwrap rsa-aes comando usando a chave privada RSA com o valor do `unwrap` atributo `true` definido como.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados 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
+NcerbAJO8DJ1lXPygZcI21vIFSZJuWMEiWpe1R9D/5WSYgxLVKex30xCFqebtEzxbKuv4DOmU4meSofqREYvtb3EoIKwjy
+RL5WGXKe4nAboAkC5GO7veI5yHL1SaKlssSJtTL/CFpbSLsAFuYbv/NUCWwMY5mwyVTCSlw+HlgKK
+5TH1MzBaSi8fpfyepLT8sHy2Q/VRl6ifb49p6m0KQFbRVvz/OWUd6l4d97BdgtaEz6ueg==
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001808e2",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001808e2",
      "key-info": {
        "key-owners": [
          {
```

```
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 no qual executar essa operação.

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 selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave 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: `aesdes3`,`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>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave unwrap rsa-oaep

O key unwrap rsa-oaep comando desempacota uma chave de carga usando a chave privada RSA e o mecanismo de desempacotamento. `RSA-OAEP`

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap rsa-oaep comando usando a chave privada RSA com o valor do `unwrap` atributo `true` definido como.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados 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
 OjJe4msobPLz9TuSAdULEu17T5rMDWtSlLyBSkLbaZnYzzpdrhsbGLbwZJCtB/jGkDNdB4qyTAOQwEpggGf6v
+Yx6JcesNeKKNU8XZal/YBoHC8noTGUSDI2qr+u2tDc84NPv6d+F2KOONXsSxMhmxzzNG/
gzTVIJhOuy/B1yHjGP4mOXoDZf5+7f5M1CjxBmz4Vva/wrWHGCSG0yOaWblEvOiHAIt3UBdyKmU+/
My4xjfJv7WGGu3DFUUIZ06TihRtKQhUYU1M9u6NPf9riJJfHsk6QCuSZ9yWThDT9as6i7e3htnyDhIhGWaoK8JU855cN/
YNKAUqkNpC4FPL3iw==
{
  "data": {
    "key": {
      "key-reference": "0x00000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
```

```
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 no qual executar essa operação.

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 selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave 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: `aesdes3`,`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>

Etiqueta para a chave aberta.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

chave unwrap rsa-pkcs

O key unwrap rsa-pkcs comando desempacota uma chave de carga usando a chave privada RSA e o mecanismo de desempacotamento. `RSA-PKCS`

As chaves desembrulhadas podem ser usadas da mesma forma que as chaves geradas pelo AWS CloudHSM. Para indicar que eles não foram gerados localmente, seu `local` atributo 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 criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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
  -h, --help
          Print help
```

Exemplos

Esses exemplos mostram como usar o key unwrap rsa-oaep comando usando uma chave AES com o valor do `unwrap` atributo definido como`true`.

Example Exemplo: Desempacotar uma chave de carga útil dos dados da chave encapsulada codificados 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+5HvU7sIBcXVb24QA0l65nbNAD+1bK+e18BpSfnaI3P+r8Dp+pLu1ofoUy/
vtzRjZoCiDofcz4EqCFnGl4GdcJ1/3W/5WRvMatCa2d7cx02swaeZcjKsermPXYRO1lGlfq6NskwMeeTkV8R7Rx9artFrs1
c3XdFJ2+0Bo94c6og/
yfPcpOOobJlITCoXhtMRepSdO4OggYq/6nUDuHCtJ86pPGnNahyr7+sAaSI3a5ECQLUjwaIARUCyoRh7EFK3qPXcg==
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08ef",
      "key-info": {
        "key-owners": [
          {
```

```
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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: Desembrulhe uma chave de carga 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": "0x00000000001c08ef",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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 no qual executar essa operação.

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 selecionar uma chave com `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` a qual desempacotar.

Obrigatório: Sim

<DATA_PATH>

Caminho para o arquivo binário contendo os dados da chave encapsulada.

Obrigatório: Sim (a menos que seja fornecido por meio de dados codificados em Base64)

<DATA>

Dados-chave agrupados codificados em Base64.

Obrigatório: Sim (a menos que seja fornecido por meio do caminho de dados)

<ATTRIBUTES>

Lista separada por espaço de atributos de chave 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: `aesdes3`,`ec-private`,,`generic-secret`,`rsa-private`].

Obrigatório: Sim

<LABEL>

Etiqueta para a chave desembrulhada.

Obrigatório: Sim

<SESSION>

Cria uma chave de sessão que existe somente na sessão atual. A chave não pode ser recuperada após o término da sessão.

Obrigatório: não

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves

O key wrap comando na CLI do CloudHSM exporta uma cópia criptografada de uma chave privada simétrica ou assimétrica do HSM para um arquivo. Ao executarkey wrap, você especifica duas coisas: a chave para exportar 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 key wrap comando não remove a chave do HSM nem impede que você a use em operações criptográficas. É possível exportar a mesma chave várias vezes. Para importar a chave criptografada de volta para o HSM, usechave: desembrulhar. Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. Os usuários com quem a chave é compartilhada só podem usá-la em operações criptográficas.

O key wrap comando consiste nos seguintes subcomandos:

- envoltório de chaves aes-gcm
- envoltório de chaves aes-no-pad
- envoltório de chaves aes-pkcs5-pad
- envoltório de chaves aes-zero-pad
- envoltório de chaves cloudhsm-aes-gcm
- envoltório de chaves rsa-aes
- envoltório de chaves rsa-oaep
- envoltório de chaves rsa-pkcs

envoltório de chaves aes-gcm

O key wrap aes-gcm comando agrupa uma chave de carga usando uma chave AES no HSM e o mecanismo de encapsulamentoAES-GCM. O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 empacotar com o key generate-symmetric aes comando e o `wrap` atributo `true` definidos como.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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 key wrap aes-gcm comando 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": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "iv": "0xf90613bb8e337ec0339aad21",
    "wrapped_key_data": "xvslgrtg8kHzrvekny97tLSIeokpPwV8"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

Obrigatório: se vários clusters tiverem sido configurados.

<PAYLOAD_FILTER>

Referência de chave (por exemplo,`key-reference=0xabc`) ou lista separada por espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

Obrigatório: Sim

<AAD>

Valor de dados autenticados adicionais (AAD) do AES GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Comprimento da etiqueta AES GCM em bits.

Obrigatório: Sim

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves aes-no-pad

O key wrap aes-no-pad comando agrupa uma chave de carga usando uma chave AES no HSM e o mecanismo de empacotamento`AES-NO-PAD`. O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 empacotar usando o key generate-symmetric aes comando e o `wrap` atributo `true` definido como.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Este exemplo mostra como usar o key wrap aes-no-pad comando usando uma chave AES com o valor do `wrap` atributo 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": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "eXK3PMAOnKM9y3YX6brbhtMoC060EOH9"
  }
}
```

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 espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

Obrigatório: Sim

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves aes-pkcs5-pad

O key wrap aes-pkcs5-pad comando agrupa uma chave de carga usando uma chave AES no HSM e o mecanismo de empacotamentoAES-PKCS5-PAD. O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 empacotar usando o key generate-symmetric aes comando e o `wrap` atributo `true` definido como.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Este exemplo mostra como usar o key wrap aes-pkcs5-pad comando usando uma chave AES com o valor do `wrap` atributo 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": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "MbuYNresfOKyGNnxKWen88nSfX+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 espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

Obrigatório: Sim

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves aes-zero-pad

O key wrap aes-zero-pad comando agrupa uma chave de carga usando uma chave AES no HSM e o mecanismo de empacotamento`AES-ZERO-PAD`. O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 empacotamento usando o key generate-symmetric aes comando com o `wrap` atributo `true` definido como.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Este exemplo mostra como usar o key wrap aes-zero-pad comando usando uma chave AES com o valor do `wrap` atributo 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": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "L1wVlL/YeBNVAw6Mpk3owFJZXBzDLONt"
  }
}
```

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 espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

Obrigatório: Sim

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves cloudhsm-aes-gcm

O key wrap cloudhsm-aes-gcm comando encapsula uma chave de carga útil usando uma chave AES no HSM e o mecanismo de encapsulamento`CLOUDHSM-AES-GCM`. O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 empacotar com o key generate-symmetric aes comando e o `wrap` atributo `true` definidos como.

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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 key wrap cloudhsm-aes-gcm comando 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": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "6Rn8nkjEriDYlnP3P8nPkYQ8hplOEJ899zsrF+aTB0i/fIlZ"
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

Obrigatório: se vários clusters tiverem sido configurados.

<PAYLOAD_FILTER>

Referência de chave (por exemplo,`key-reference=0xabc`) ou lista separada por espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

Obrigatório: Sim

<AAD>

Valor de dados autenticados adicionais (AAD) do AES GCM, em hexadecimal.

Obrigatório: não

<TAG_LENGTH_BITS>

Comprimento da etiqueta AES GCM em bits.

Obrigatório: Sim

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves rsa-aes

O key wrap rsa-aes comando agrupa uma chave de carga usando uma chave pública RSA no HSM e o mecanismo de encapsulamento RSA-AES. O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 chave generate-asymmetric-pair comando e o `wrap` atributo definido como. `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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 key wrap rsa-ae comando usando uma chave pública RSA com o valor do `wrap` atributo `true` definido como.

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": "0x00000000001c08f1",
    "wrapping-key-reference": "0x00000000007008da",
    "wrapped-key-data": "HrSE1DEyLjIeyGdPa9R+ebiqB5TIJGyamPker31ZebPwRA
+NcerbAJO8DJ1lXPygZcI21vIFSZJuWMEiWpe1R9D/5WSYgxLVKex30xCFqebtEzxbKuv4DOmU4meSofqREYvtb3EoIKwjy
+RL5WGXKe4nAboAkC5GO7veI5yHL1SaKlssSJtTL/CFpbSLsAFuYbv/NUCWwMY5mwyVTCSlw+HlgKK
+5TH1MzBaSi8fpfyepLT8sHy2Q/VRl6ifb49p6m0KQFbRVvz/OWUd6l4d97BdgtaEz6ueg=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

Obrigatório: se vários clusters tiverem sido configurados.

<PAYLOAD_FILTER>

Referência de chave (por exemplo,`key-reference=0xabc`) ou lista separada por espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

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

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves rsa-oaep

O key wrap rsa-oaep comando agrupa uma chave de carga usando uma chave pública RSA no HSM e o mecanismo de agrupamento. `RSA-OAEP` O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 chave generate-asymmetric-pair comando e o `wrap` atributo definido como. `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
      --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 key wrap rsa-oaep comando usando uma chave pública RSA com o valor do `wrap` atributo `true` definido como.

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": "0x00000000001c08f1",
    "wrapping-key-reference": "0x00000000007008da",
    "wrapped-key-data": "OjJe4msobPLz9TuSAdULEu17T5rMDWtSlLyBSkLbaZnYzzpdrhsbGLbwZJCtB/
jGkDNdB4qyTAOQwEpggGf6v+Yx6JcesNeKKNU8XZal/YBoHC8noTGUSDI2qr+u2tDc84NPv6d
+F2KOONXsSxMhmxzzNG/gzTVIJhOuy/B1yHjGP4mOXoDZf5+7f5M1CjxBmz4Vva/
wrWHGCSG0yOaWblEvOiHAIt3UBdyKmU+/
My4xjfJv7WGGu3DFUUIZ06TihRtKQhUYU1M9u6NPf9riJJfHsk6QCuSZ9yWThDT9as6i7e3htnyDhIhGWaoK8JU855cN/
YNKAUqkNpC4FPL3iw=="
  }
}
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

Obrigatório: se vários clusters tiverem sido configurados.

<PAYLOAD_FILTER>

Referência de chave (por exemplo,`key-reference=0xabc`) ou lista separada por espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

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

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

envoltório de chaves rsa-pkcs

O key wrap rsa-pkcs comando agrupa uma chave de carga usando uma chave pública RSA no HSM e o mecanismo de agrupamento. `RSA-PKCS` O `extractable` atributo da chave de carga útil deve ser definido como. `true`

Somente o proprietário de uma chave, ou seja, o usuário criptográfico (UC) que criou a chave, pode encapsular a chave. 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 chave generate-asymmetric-pair comando e o `wrap` atributo definido como. `true`

Tipo de usuário

Os seguintes tipos de usuários podem executar este comando.

- Usuários de criptografia (CUs)

Requisitos

- Para executar esse comando, você deve estar registrado 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
  -h, --help
          Print help
```

Exemplo

Este exemplo mostra como usar o key wrap rsa-pkcs comando 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": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000007008da",
    "wrapped_key_data": "am0Nc7+YE8FWs+5HvU7sIBcXVb24QA0l65nbNAD+1bK+e18BpSfnaI3P+r8Dp
+pLu1ofoUy/
vtzRjZoCiDofcz4EqCFnGl4GdcJ1/3W/5WRvMatCa2d7cx02swaeZcjKsermPXYRO1lGlfq6NskwMeeTkV8R7Rx9artFrs1
c3XdFJ2+0Bo94c6og/
yfPcpOOobJlITCoXhtMRepSdO4OggYq/6nUDuHCtJ86pPGnNahyr7+sAaSI3a5ECQLUjwaIARUCyoRh7EFK3qPXcg=="
  }
```

Argumentos

<CLUSTER_ID>

O ID do cluster no qual executar essa operação.

Obrigatório: se vários clusters tiverem sido configurados.

<PAYLOAD_FILTER>

Referência de chave (por exemplo,`key-reference=0xabc`) ou lista separada por espaço 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 espaço de atributos de chave na forma de `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` selecionar uma chave de empacotamento.

Obrigatório: Sim

Tópicos relacionados

- envoltório de chaves
- chave: desembrulhar

login

Você pode usar o comando login na CLI do CloudHSM para fazer login e logout de cada HSM em um cluster.

> **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 pode ocorrer porque o cliente do CloudHSM divide a carga entre 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
- Usuário de criptografia (CU)

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 faz login em todos os HSMs em um cluster com as credenciais de um administrador denominado `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 no qual executar essa operação.

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 fez login nos HSMs.

Tópicos relacionados da

- Conceitos básicos sobre a CLI do CloudHSM
- Ativar o cluster

logar mfa-token-sign

Use o login mfa-token-sign comando no log da CLI do CloudHSM AWS CloudHSM usando autenticação multifatorial. Para usar esse comando, você precisa primeiro configurar a MFA para a CLI do CloudHSM.

Tipo de usuário

Os usuários a seguir podem executar esses comandos.

- Administrador
- Usuário de criptografia (CU)

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 da

- Conceitos básicos sobre a CLI do CloudHSM
- Ativar o cluster
- Usar a CLI do CloudHSM para gerenciar a MFA

logout

Você pode usar o comando logout CloudHSM CLI (CLI do CloudHSM) para fazer logout de cada HSM em um cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Usuário de criptografia (CU)

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 todos os HSMs em um cluster.

```
aws-cloudhsm > logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

Tópicos relacionados da

- Conceitos básicos sobre a CLI do CloudHSM
- Ativar o cluster

usuário

user é uma categoria principal para um grupo de comandos que, quando combinados com a categoria principal, criam um comando específico para os usuários. Atualmente, a categoria de usuário consiste nos seguintes comandos:

- user change-mfa
- alterar senha de usuário
- criar usuário
- excluir usuário
- ista de usuários

user change-mfa

Atualmente, essa categoria consiste nos seguintes subcomandos:

- sinal de token de mfa de mudança de usuário

sinal de token de mfa de mudança de usuário

Use o user change-mfa comando 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 no qual executar essa operação.

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 da

- Noções básicas sobre 2FA para usuários de HSM

alterar senha de usuário

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
- Usuário de criptografia (CU)

Sintaxe

> **Note**
>
> Para habilitar a autenticação multifatorial (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 no qual executar essa operação.

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.

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 da

- lista de usuários
- criar usuário
- excluir usuário

quórum de mudança de usuário

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 quorum para os usuários.

user change-quorumé 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

sinal de token de quorum de mudança de usuário

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

user change-quorum token-sign register

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 <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-key <PUBLIC_KEY>      Filepath to public key PEM file
      --signed-token <SIGNED_TOKEN>  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 no qual executar essa operação.

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 da

- 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

criar usuário

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

Estes 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 fez login nos HSMs.

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 da

- lista de usuários
- excluir usuário
- alterar senha de usuário

excluir usuário

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 no qual executar essa operação.

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 da

- lista de usuários
- criar usuário
- alterar senha de usuário

lista de usuários

O comando user list na CLI do CloudHSM CLI lista as contas de usuário presentes no seu cluster do 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 que você fizer podem não ser efetivas em todos os 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

```
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 da

- listUsers em key_mgmt_util
- criar usuário
- excluir usuário

- alterar senha de usuário

quorum

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.

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

Tópicos relacionados

- Usar a autenticação de quórum para administradores: configuração inicial
- Como usar o CLI do CloudHSM para gerenciar a autenticação de quórum (controle de acesso M ou N)

token de quórum assinado

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
- list
- list-quorum-values
- list-timeouts
- set-quorum-value
- set-timeout

quorum token-sign delete

Use o comando do quorum token-sign delete na CLI do CloudHSM 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 CLI do CloudHSM 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 o (s) token (s) será (ão) excluído (s) 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 da

- lista de usuários
- criar usuário
- excluir usuário

quorum token-sign generate

Use o comando quorum token-sign generate na CLI do CloudHSM 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.

> **Note**
>
> Somente administradores podem gerar um token de serviço.

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.

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

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Usuário de criptografia (CU)

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
          - registration:
            Registration service is used for registering a public key for quorum
 authentication

      --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.
- registro: gera um token não assinado para uso no registro de uma chave pública para autorização de quórum.

Obrigatório: Sim

<TOKEN>

Caminho onde o arquivo de token não assinado será gravado.

Obrigatório: Sim

Tópicos relacionados da

- Nomes e tipos de serviços que oferecem suporte à autenticação de quórum

quorum token-sign list

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

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador
- Usuário de criptografia (CU)

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

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": 2
        "token-timeout-seconds": 588
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Tópicos relacionados da

- quorum token-sign generate

sinal simbólico de quórum list-quorum-values

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

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 1
  }
}
```

Tópicos relacionados da

- Nomes e tipos de serviços que oferecem suporte à autenticação de quórum

quorum token-sign list-timeouts

Use o comando do quorum token-sign list-timeouts na CLI do CloudHSM para obter o período de tempo limite do token em segundos para todos os tipos de token.

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-timeouts
List timeout durations in seconds for token validity

Usage: quorum token-sign list-timeouts

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

```
aws-cloudhsm > quorum token-sign list-timeouts
{
  "error_code": 0,
  "data": {
    "generated": 600,
    "approved": 600
  }
}
```

A saída inclui a linha a seguir:

- gerado: período de tempo limite em segundos para que um token gerado seja aprovado.
- aprovado: período de tempo limite em segundos para que um token aprovado seja usado para executar uma operação autorizada por quórum.

Tópicos relacionados da

- quorum token-sign set-timeout

sinal simbólico de quórum set-quorum-value

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

      --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:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 2
  }
}
```

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.
- 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 da

- sinal simbólico de quórum list-quorum-values
- Nomes e tipos de serviços que oferecem suporte à autenticação de quórum

quorum token-sign set-timeout

Use o comando do quorum token-sign set-timeout na CLI do CloudHSM para definir o período de tempo limite do token em segundos para todos os tipos de token.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Administrador

Sintaxe

```
aws-cloudhsm > help quorum token-sign set-timeout
Set timeout duration in seconds for token validity

Usage: quorum token-sign set-timeout <--generated <GENERATED> |--approved <APPROVED>>

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
      --generated <GENERATED>     Timeout period in seconds for a generated (non-
approved) token to be approved
      --approved <APPROVED>      Timeout period in seconds for an approved token to be
 used to execute a quorum operation
  -h, --help    Print help (see a summary with '-h')
```

Exemplo

O exemplo a seguir mostra como usar o comando quorum token-sign set-timeout para obter o período de tempo limite do token.

```
aws-cloudhsm > quorum token-sign set-timeout --generated 900
{
  "error_code": 0,
```

```
  "data": "Set token timeout successful"
}
```

Tópicos relacionados da

- quorum token-sign list-timeouts

CloudHSM Management Utility (CMU – utilitário de gerenciamento do CloudHSM)

A ferramenta da linha de comando da cloudhsm_mgmt_util ajuda os responsáveis pela criptografia a gerenciar usuários nos HSMs. Inclui ferramentas que criam, excluem e listam usuários e alteram senhas de usuários.

O KMU e o CMU 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 HSM hsm1.medium.

cloudhsm_mgmt_util também inclui comandos que permitem que usuários de criptografia (CUs) compartilhem chaves e obtenham e configurar 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 Gerenciando clusters clonados. Para obter informações detalhadas sobre comandos da cloudhsm_mgmt_util e exemplos de uso dos comandos, consulte referência do comando cloudhsm_mgmt_util.

Tópicos

- Plataformas suportadas para o utilitário AWS CloudHSM de gerenciamento
- Introdução ao CloudHSM Management Utility (CMU)
- Instalar e configurar o AWS CloudHSM cliente (Linux)
- Instalar e configurar o AWS CloudHSM cliente (Windows)
- referência do comando cloudhsm_mgmt_util

Plataformas suportadas para o utilitário AWS CloudHSM de gerenciamento

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 CloudHSM Management Utility (CMU)

O CloudHSM Management Utility (CMU) permite que você gerencie 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 de configuração para atualizar a configuração local do CMU com o parâmetro `--cmu` e um endereço IP de um dos HSMs no seu 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

```
C:\Program Files\Amazon\CloudHSM\bin\ 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

   ```
   C:\Program Files\Amazon\CloudHSM> .\cloudhsm_mgmt_util.exe C:\ProgramData\Amazon
   \CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

   A saída deve ser semelhante ao seguinte, dependendo de quantos 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 CO 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 no 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              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
Users on server 1(10.0.3.11):
Number of users found:2

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
Users on server 2(10.0.1.12):
Number of users found:2

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
```

```
        1              CO             admin                          NO
             0              NO
        2              AU             app_user                       NO
             0              NO
```

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
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
         3              CU              example_user                            NO
               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              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
         3              CU              example_user                            NO
               0               NO
Users on server 2(10.0.1.12):
Number of users found:3

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
         3              CU              example_user                            NO
               0               NO
```

8. Use o comando logoutHSM para fazer logout dos 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...
```

Instalar e configurar o AWS CloudHSM cliente (Linux)

Para interagir com o HSM em seu AWS CloudHSM cluster, você precisa do software AWS CloudHSM cliente para Linux. 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 ter mais informações, consulte Instalar e configurar o AWS CloudHSM cliente (Windows).

Tarefas

- Instale as ferramentas AWS CloudHSM do cliente e da linha de comando
- Editar a configuração do cliente

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
```

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 config -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 na 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`. Caso tenha mais de um HSM, selecione o endereço IP para qualquer um dos 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 Ativar o cluster.

Instalar e configurar o AWS CloudHSM cliente (Windows)

Para trabalhar com um HSM em seu AWS CloudHSM cluster no Windows, você precisa do software AWS CloudHSM cliente para Windows. 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. Faça o download do instalador AWSCloudHSMClient -latest.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 -a ***<IP of one of the HSMs>***.
   b. Reinicie o serviço do cliente.
   c. Executar config -m.
4. Vá para 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 clusterpara 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:

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

8. Acesse Ativar o cluster.

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 ter mais informações, consulte Instalar e configurar o AWS CloudHSM cliente (Linux).

referência do comando cloudhsm_mgmt_util

A ferramenta da linha de comando da cloudhsm_mgmt_util ajuda os responsáveis pela criptografia a gerenciar usuários nos HSMs. Ela também inclui comandos que permitem que usuários de criptografia (CUs) compartilhem chaves, além de obter e configurar 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 Gerenciando clusters clonados.

Antes de executar um comando cloudhsm_mgmt_util, você deve iniciar cloudhsm_mgmt_utile 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 nos HSMs, insira 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 Referência de atributos de chave.

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	Descrição	Tipo de usuário
changePswd	Altera as senhas dos usuários nos HSMs. Qualquer usuário pode alterar sua própria senha. Os COs podem mudar a senha de qualquer pessoa.	CO
createUser	Cria usuários de todos os tipos nos HSMs.	CO
deleteUser	Exclui usuários de todos os tipos dos HSMs.	CO
findAllKeys	Obtém as chaves que um usuário possui ou compartil ha. Também obtém um hash de propriedade da chave e compartilhamento de dados	CO, AU

Command	Descrição	Tipo de usuário
	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
getCert	Obtenha o certificado de um HSM específico e o salve em um formato de certificado desejado.	Tudo.
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 de usuário e ID 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.

Command	Descrição	Tipo de usuário
servidor	Entra e sai do modo de servidor em um HSM.	Tudo.
registerQuorumPubChave	Associa um usuário do HSM a um par de chaves RSA-2048 assimétrico.	CO
setAttribute	Altera o valor dos atributos de rótulo, criptografia, descripto grafia, encapsulamento e desencapsulamento de uma chave existente.	CU
shareKey	Compartilha uma chave existente com outros usuários.	CU
syncKey	Sincroniza uma chave em clusters clonados AWS CloudHSM .	CU, CO
syncUser	Sincroniza um usuário em clusters clonados AWS CloudHSM .	CO

changePswd

O comando changePswd em cloudhsm_mgmt_util altera a senha de um usuário existente nos HSMs do cluster.

Qualquer usuário pode alterar sua própria senha. Além disso, os Crypto officers (responsáveis pela criptografia) (COs e PCOs) podem alterar a senha de outro CO ou usuário de criptografia (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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 ter 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 seus HSMs.

Example : Alterar sua senha

Qualquer usuário nos HSMs pode usar changePswd para alterar sua própria senha. Antes de alterar a senha, use info para obter informações sobre cada um dos HSMs no cluster, incluindo o nome e o tipo de 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 um CO ou PCO para alterar a senha de outro CO ou CU nos 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(CO)'


aws-cloudhsm>info server 1

Id      Name             Hostname         Port   State           Partition
 LoginState
0       10.1.10.7        10.1.10.7        2225   Connected       hsm-ogi3sywxbqx
 Logged in as 'alice(CO)'
```

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
 LoginFailureCnt         2FA
        1               PCO             admin                   YES              0
            NO
        2               AU              jane                     NO              0
            NO
        3               CU              bob                      NO              0
            NO
```

```
         4               CU              alice                    NO              0
             NO
         5               CO              john                     NO              0
             NO
Users on server 1(10.1.10.7):
Number of users found:5

    User Id             User Type       User Name            MofnPubKey
 LoginFailureCnt         2FA
         1               PCO             admin                   YES              0
             NO
         2               AU              jane                     NO              0
             NO
         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 Usando a CMU para gerenciar 2FA

```
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 Noções básicas sobre usuários do HSM.

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 Usando a CMU para gerenciar 2FA.

Obrigatório: não

Tópicos relacionados da

- info
- listUsers
- createUser

- deleteUser

createUser

O comando createUser na cloudhsm_mgmt_util cria um usuário nos HSMs. Somente responsáveis pela criptografia (COs e PRECOs) podem executar esse comando. Quando o comando é bem-sucedido, ele cria o usuário em todos os HSMs no cluster.

Para solucionar problemas de createUser

No entanto, se sua configuração do HSM estiver imprecisa, o usuário talvez não seja criado em todos os HSMs. Para adicionar o usuário a qualquer HSM em que ele esteja faltando, use o comando syncUser ou createUser apenas nos HSMs que não têm 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 ter mais informações, consulte the section called “Argumentos”.

```
createUser <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

Exemplos

Estes exemplos mostram como usar createUser para criar novos usuários em seus HSMs.

Example : Criar um responsável pela criptografia

Este exemplo cria um responsável pela criptografia (CO) nos 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 precaução explica que o comando cria usuários em todos os HSMs no cluster. Porém, se o comando falhar em qualquer HSM, o usuário não existirá nesses HSMs. Para continuar, digite `y`.

A saída mostra que o novo usuário foi criado nos três HSMs do cluster.

```
aws-cloudhsm> createUser CO 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(CO) on 3 nodes
```

Quando o comando é concluído, `alice` tem as mesmas permissões no HSM que o usuário CO `admin`, incluindo a alteração da senha de qualquer usuário nos HSMs.

O comando final usa o comando listUsers para verificar se `alice` existe nos 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 precaução, a saída mostra que bob foi criado nos três HSMs 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 Usando a CMU para gerenciar 2FA.

```
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 Noções básicas sobre usuários do HSM.

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 CO podem alterar a senha de qualquer usuário (de qualquer tipo) nos 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 Usando a CMU para gerenciar 2FA.

Obrigatório: não

Tópicos relacionados da

- listUsers
- deleteUser
- syncUser
- changePswd

deleteUser

O comando deleteUser em cloudhsm_mgmt_util exclui um usuário dos módulos de segurança de hardware (HSM). 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 responsável pela criptografia (CO) dos HSMs em um cluster. O primeiro comando usa listUsers para listar todos os usuários nos HSMs.

A saída mostra que o usuário `3`, `alice`, é uma CO nos HSMs.

```
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               PCO             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              PCO             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              PCO             admin                                    YES
          0               NO
        2              AU              app_user                                 NO
          0               NO
        3              CO              alice                                    NO
          0               NO
```

O segundo comando usa o comando deleteUser para excluir `alice` dos HSMs.

A saída mostra que o comando foi bem-sucedido em todos os três HSMs no cluster.

```
aws-cloudhsm> deleteUser CO alice
Deleting user alice(CO) 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 comando listUsers comando para verificar se `alice` foi excluída 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              PCO             admin                               YES
           0               NO
         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 Noções básicas sobre usuários 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

Tópicos relacionados da

- listUsers
- createUser
- syncUser
- changePswd

findAllKeys

O comando findAllKeys na cloudhsm_mgmt_util obtém as chaves que um determinado usuário de criptografia (CU) possui ou compartilha. Ele também retorna um hash dos dados do usuário em cada um dos HSMs. Você pode usar esse hash para determinar de imediato se os usuários, a propriedade das chaves e os dados de compartilhamento de chaves são iguais em todos os HSMs 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 CU especificada possui a chave, mesmo que todos os 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 os responsáveis pela criptografia (COs e PCOs) e usuários dos dispositivos (AUs) podem executar esse comando. Os usuários de criptografia (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
- getKeyInfoem 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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

Estes exemplos mostram como usar `findAllKeys` para encontrar todas as chaves para um usuário e obter um hash de informações de usuário de chaves em cada um dos HSMs.

Example : Encontrar as chaves para um CU

Esse exemplo usa findAllKeys para encontrar as chaves nos 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 usa findAllKeys para verificar se todos os HSMs do cluster contêm os mesmos usuários, propriedade de chaves e valores de compartilhamento de chaves. 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 usuário 6, mas o valor do hash será o mesmo para qualquer usuário que possua ou compartilhe qualquer uma das chaves nos 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 tanto nos HSMs como 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 o ID do usuário de um usuário nos HSMs. Para encontrar os IDs de todos os usuários, use listUsers.

Todos os IDs de usuários são válidos, mas `findAllKeys` retorna as chaves apenas para os usuários de criptografia (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 da

- changePswd
- deleteUser
- listUsers
- syncUser
- findKey em key_mgmt_util

- getKeyInfoem key_mgmt_util

getAttribute

O comando getAttribute na cloudhsm_mgmt_util obtém um valor de atributo para uma chave de todos os HSMs no cluster e grava-o na stdout (saída padrão) ou em um arquivo. Somente usuários de criptografia (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 Referência de atributos de 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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

Esse exemplo obtém o valor do atributo extraível para uma chave nos HSMs. É possível usar um comando como esse para determinar se você pode exportar uma chave dos 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 Referência de atributos de chave.

```
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 usa getAttribute para obter o valor do atributo extraível da chave com o identificador de chave `262170` nos 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 dos 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 getKeyInfoem 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 Referência de atributos de chave.

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 da

- getAttribute em key_mgmt_util
- listAttributes
- setAttribute em cloudhsm_mgmt_util
- setAttribute em key_mgmt_util
- Referência de atributos de chave

getCert

Com o comando getCert na cloudhsm_mgmt_util, é possível recuperar os certificados de um HSM específico em um cluster. Ao executar o comando, você designa o tipo de certificado a ser recuperado. Para fazer isso, use um dos inteiros correspondentes, conforme descrito na seção Argumentos abaixo. Para saber sobre a função de cada um desses certificados, consulte Verificar a identidade do 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os HSMs no cluster.

Tipo de usuário

Os usuários a seguir podem executar este comando.

- Todos os usuários.

Pré-requisitos

Antes de começar, você deve inserir o modo de servidor no HSM de destino. Para obter mais informações, consulte servidor.

Sintaxe

Para usar o comando getCert depois que estiver em modo de servidor:

```
server> getCert <file-name> <certificate-type>
```

Exemplo

Primeiro, insira o modo de servidor. Esse comando insere o modo de servidor em um HSM com o servidor número `0`.

```
aws-cloudhsm> server 0

Server is in 'E2' mode...
```

Depois, use o comando getCert. Neste exemplo, usamos `/tmp/P0.crt` como o nome do arquivo no qual o certificado será salvo e `4` (Certificado raiz do cliente) como o tipo de certificado desejado:

```
server0> getCert /tmp/P0.crt 4
getCert Success
```

Argumentos

```
getCert <file-name> <certificate-type>
```

<file-name>

Especifica o nome do arquivo no qual o certificado será salvo.

Obrigatório: Sim

<certificate-type>

Um inteiro que especifica o tipo de certificado 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 da

- servidor

getHSMInfo

O comando getHSMInfo na cloudhsm_mgmt_util obtém informações sobre o hardware no qual cada HSM é executado, incluindo o modelo, o número de série, o estado FIPS, a memória, a 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 usa getHSMInfo para obter informações sobre os 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          :xxxxxxxxxxxxxx

...
```

Tópicos relacionados da

- info

getKeyInfo

O comando getKeyInfo na key_mgmt_util retorna os IDs de usuário do HSM dos usuários que podem usar a chave, incluindo o proprietário e os usuários de criptografia (CU) com quem 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 localizar os IDs de usuário do HSM nos seus HSMs, use listUsers. Para encontrar as chaves de um usuário específico, use findKey -u na key_mgmt_util. Os agentes de criptografia podem usar findAllKeysem 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 executa getKeyInfo na chave privada (262177), ele retorna o proprietário da chave (3) e os usuários de criptografia (CUs) 4, com quem 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 genSymKeypara 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 da

- getKeyInfoem key_mgmt_util
- findKey em key_mgmt_util
- findAllKeysem cloudhsm_mgmt_util
- listUsers
- shareKey

info

O info comando em cloudhsm_mgmt_util obtém informações sobre cada um dos HSMs no cluster, incluindo o nome do host, a porta, o endereço IP e o nome e tipo do usuário que está conectado a 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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. Os HSMs recebem números ordinais que representam a ordem em que são adicionados ao cluster, começando com 0. Para encontrar o ID de servidor de um HSM, use getHSMInfo.

Obrigatório: Sim

Tópicos relacionados da

- getHSMInfo
- loginHSM e logoutHSM

listAttributes

O listAttributes comando em cloudhsm_mgmt_util lista 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 Referência de atributos de chave.

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 Referência de atributos de chave. 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 da

- getAttribute
- setAttribute
- Referência de atributos de chave

listUsers

O comando listUsers na cloudhsm_mgmt_util obtém os usuários em cada um dos HSMs, juntamente com seu 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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              PCO             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               CO              officer1                    YES             0
            NO
        6               CO              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               PCO             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               CO              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 da

- listUsers em key_mgmt_util
- createUser
- deleteUser
- changePswd

loginHSM e logoutHSM

Você pode usar os comandos loginHSM e logoutHSM no cloudhsm_mgmt_util 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 que você fizer podem não ser efetivas em todos os HSMs no 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 pode ocorrer porque o cliente do CloudHSM divide a carga entre 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)
- Usuário de criptografia (CU)

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 ter 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 para fazer login e logout de todos os HSMs em um cluster.

Example : fazer login no HSMs em um cluster

Esse comando faz login em todos os HSMs em um cluster com as credenciais de um usuário CO `admin` e uma senha de `co12345`. O resultado mostra que o comando foi bem-sucedido e que o usuário se conectou aos HSMs (que, neste caso, são `server 0` e `server 1`).

```
aws-cloudhsm>loginHSM CO admin co12345

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 CO 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 aos 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 faz logout dos HSMs aos quais você está conectado no momento (que, neste caso, são `server 0` e `server 1`). O resultado mostra que o comando foi bem-sucedido e que o usuário se desconectou dos 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 Usando a CMU para gerenciar 2FA

```
loginHSM <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

<user type>

Especifica o tipo de usuário que fez login nos HSMs. Para obter mais informações, consulte Tipo de usuário acima.

Obrigatório: Sim

<user name>

Especifica o nome do usuário que fez login nos HSMs.

Obrigatório: Sim

<password | -hpswd >

Especifica a senha do usuário que fez login nos 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 Usando a CMU para gerenciar 2FA.

Obrigatório: não

Tópicos relacionados da

- Conceitos básicos de cloudhsm_mgmt_util
- Ativar o cluster

registerQuorumPubChave

O comando registerQuorumPubKey em cloudhsm_mgmt_util associa 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 : registre 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               PCO             admin                           NO
          0               NO
        2               AU              app_user                        NO
          0               NO
        3               CO              quorum_officer                  YES
          0               NO
```

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 Noções básicas sobre usuários do HSM.

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 da

- 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

servidor

Normalmente, ao emitir um comando na cloudhsm_mgmt_util, o comando afeta todos os 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. É possível usar o comando server na cloudhsm_mgmt_util para entrar no modo de servidor e interagir diretamente com uma instância específica do HSM.

Após o início bem-sucedido, o prompt de comando `aws-cloudhsm>` será substituído pelo prompt de comando `server>`.

Para 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 os 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 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 da

- syncKey
- createUser
- deleteUser

setAttribute

O comando setAttribute na cloudhsm_mgmt_util altera o valor dos atributos de rótulo, criptografia, descriptografia, agrupamento e desagrupamento de uma chave nos 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 genSymKeyem 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 Referência de atributos de chave.

```
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
```

Para obter o valor atual do atributo de descriptografia para a chave 14, o próximo comando getAttribute na cloudhsm_mgmt_util.

A saída mostra que o valor do atributo decrypt é true (1) em ambos os 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 descriptografia (atributo 261) da chave 14 para 0. Isso desabilita a funcionalidade de descriptografia 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 de descriptografia (atributo 261) da chave 14.

Dessa vez, a saída mostra que o valor do atributo de descriptografia é false (0) em ambos os 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 uma atributo em cada comando. Para obter os atributos e seus valores inteiros, use listAttributes. Para obter ajuda sobre a interpretação dos principais atributos, consulte Referência de atributos de chave.

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 da

- setAttribute em key_mgmt_util
- getAttribute
- listAttributes
- Referência de atributos de chave

sair

O comando quit no cloudhsm_mgmt_util sai 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 da

- Conceitos básicos de cloudhsm_mgmt_util

shareKey

O comando shareKey na cloudhsm_mgmt_util compartilha e descompartilha 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 de criptografia 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 getKeyInfopara obter as informações do usuário para a chave262177, uma chave privada ECC nos 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 exemplo usa shareKey para compartilhar a chave 262177 com o usuário 4, outro usuário de criptografia nos 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 os 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 4da 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 os 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

syncKey

Você pode usar o comando syncKey na cloudhsm_mgmt_util para sincronizar os usuários entre as instâncias do HSM em um cluster ou entre 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 usarsyncKey, 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**
>
> Os COs podem usar syncKey em qualquer chave; enquanto os CUs só podem usar esse comando em chaves que possuem. Para ter mais informações, consulte the section called "Noções básicas sobre usuários do HSM".

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 findAllKeyscomando para encontrar todas as chaves que pertencem a um determinado usuário.

Você também precisa saber o `server IDs` atribuído aos HSMs de origem e de destino, que são mostrados na saída do rastreamento retornado pelo cloudhsm_mgmt_util na inicialização. Eles são atribuídos na mesma ordem em que os 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 chave de uma chave, use findAllKeysenquanto 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 da

- listUsers
- findAllKeys
- descreva-clusters em AWS CLI
- servidor

syncUser

Você pode usar o syncUser comando em cloudhsm_mgmt_util para sincronizar manualmente usuários criptográficos (CUs) ou agentes criptográficos (CoS) em instâncias de HSM em um cluster ou em clusters clonados. AWS CloudHSM não sincroniza automaticamente os usuários. Em geral, você gerencia usuários no modo global para que todos os HSMs em um cluster sejam atualizados em conjunto. 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 adicionar ou excluir HSMs, atualize os arquivos de configuração do CMU. Caso contrário, as alterações que você fizer podem não ser efetivas em todos os 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 localizar o `user ID`, use o comando listUsers para listar todos os usuários nos HSMs em um cluster.

Você também precisa saber o `server ID` atribuído aos HSMs de origem e de destino, que são mostrados na saída do rastreamento retornado pelo cloudhsm_mgmt_util na inicialização. Eles são atribuídos na mesma ordem em que os HSMs aparecem no arquivo de configuração.

Se você estiver sincronizando HSMs entre clusters clonados, siga as instruções em Uso do CMS entre 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 da

- listUsers
- descreve-clusters em AWS CLI
- servidor

Key Management Utility (KMU – utilitário de gerenciamento de chaves)

Use a ferramenta de linha de comando do utilitário de gerenciamento de chaves (KMU) que ajuda os usuários de criptografia (CU) 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 Conceitos básicos de key_mgmt_util. Para obter informações detalhadas sobre os comandos, consulte referência do comando key_mgmt_util. Para obter ajuda sobre a interpretação dos principais atributos, consulte Referência de atributos de chave.

Para usar key_mgmt_util se você estiver usando Linux, conecte-se à instância do cliente e, em seguida, consulte Instalar e configurar o AWS CloudHSM cliente (Linux). Se você estiver usando Windows, consulte Instalar e configurar o AWS CloudHSM cliente (Windows).

Tópicos

- Conceitos básicos de key_mgmt_util
- Instalar e configurar o AWS CloudHSM cliente (Linux)
- Instalar e configurar o AWS CloudHSM cliente (Windows)
- referência do comando key_mgmt_util

Conceitos básicos de 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 a key_mgmt_util
- Utilização básica de 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 do comando key_mgmt_util.

Configurar a key_mgmt_util

Execute a configuração a seguir antes de usar a key_mgmt_util.

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 os HSMs no seu cluster. A ferramenta key_mgmt_util usa a conexão do cliente para se comunicar com os HSMs em seu cluster. 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
```

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

```
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.

Utilização básica de key_mgmt_util

Consulte os tópicos a seguir em relação à utilização básica da ferramenta key_mgmt_util.

Tópicos

- Fazer login em HSMs
- Fazer logout nos HSMs
- Parar key_mgmt_util

Fazer login em HSMs

Use o comando loginHSM para fazer login nos HSMs. O comando a seguir efetua login como um usuário de criptografia (CU) chamado `example_user`. A saída indica um login realizado com sucesso 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>
```

Fazer logout nos HSMs

Use o comando logoutHSM para fazer logout nos HSMs.

```
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
```

Parar key_mgmt_util

Use o comando exit para parar a key_mgmt_util.

```
Command:  exit
```

Instalar e configurar o AWS CloudHSM cliente (Linux)

Para interagir com o HSM em seu AWS CloudHSM cluster, você precisa do software AWS CloudHSM cliente para Linux. 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 ter mais informações, consulte Instalar e configurar o AWS CloudHSM cliente (Windows).

Tarefas

- Instale as ferramentas AWS CloudHSM do cliente e da linha de comando
- Editar a configuração do cliente

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
```

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`. Caso tenha mais de um HSM, selecione o endereço IP para qualquer um dos 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 Ativar o cluster.

Instalar e configurar o AWS CloudHSM cliente (Windows)

Para trabalhar com um HSM em seu AWS CloudHSM cluster no Windows, você precisa do software AWS CloudHSM cliente para Windows. 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 o 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 clusterpara 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:

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

7. Acesse Ativar o cluster.

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 ter mais informações, consulte Instalar e configurar o AWS CloudHSM cliente (Linux).

referência do comando key_mgmt_util

A ferramenta de linha de comando key_mgmt_util ajuda você a gerenciar chaves nos HSMs no seu cluster, incluindo criar, excluir e encontrar chaves e seus atributos. Ela inclui vários comandos, cada um descrito em detalhes neste tópico.

Para começar rapidamente, consulte Conceitos básicos de key_mgmt_util. Para obter ajuda sobre a interpretação dos principais atributos, consulte Referência de atributos de chave. 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 CloudHSM Management Utility (CMU – utilitário de gerenciamento do 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).

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	Descrição
aesWrapUnwrap	Criptografa e descriptografa o conteúdo de uma chave em um arquivo.
deleteKey	Exclui uma chave dos 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.
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.

Command	Descrição
findKey	Procure chaves por valor de atributo de chave.
findSingleKey	Verifica se existe uma chave em todos os HSMs no cluster.
GendSA KeyPair	Gera um par de chaves de Algoritmo de assinatura digital (DSA) em seus HSMs.
GeneCC KeyPair	Gera um par de chaves de Criptografia de curva elíptica (ECC) nos seus HSMs.
Gênero A KeyPair	Gera um par de chaves assimétricas RSA nos seus HSMs.
genSymKey	Gera uma chave simétrica nos 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.
getKeyInfo	Obtém os IDs de usuários do HSM que podem usar a chave. Se a chave for controlada por quorum, ele obtém o número de usuários no quorum.
help	Exibe informações de ajuda sobre os comandos disponíveis na cloudhsm_mgmt_util.
importPrivateKey	Importa uma chave privada para um HSM.
importPubKey	Importa uma chave pública para um HSM.

Command	Descrição
imSymKey	Importa uma cópia em texto simples de uma chave simétrica de um arquivo para o HSM.
insertMaskedObject	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 gerados a partir de um backup do cluster de origem.
???	Determina se um determinado arquivo contém ou não uma chave privada real ou uma chave PEM falsa.
listAttributes	Lista os atributos de uma AWS CloudHSM chave e as constantes que os representam.
listUsers	Obtém os usuários nos HSMs, seu tipo de usuário e ID e outros atributos.
loginHSM e logoutHSM	Faça login e logout de HSMs em um cluster.
setAttribute	Converte uma chave de sessão em uma chave persistente.
sign	Gere uma assinatura para um arquivo usando uma chave privada escolhida.
unWrapKey	Importa uma chave encapsulada (criptogr afada) de um arquivo para os HSMs.
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.

aesWrapUnwrap

O comando aesWrapUnwrap criptografa ou descriptografa 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.

aesWrapUnwrap usa AES Key Wrap. 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 imSymKeycomando 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 empacotamento. 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 genSymKeypara 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 da

- exSymKey
- imSymKey
- unWrapKey
- wrapKey

deleteKey

O comando deleteKey em key_mgmt_util exclui uma chave do HSM. 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

Estes exemplos mostram como usar deleteKey para excluir chaves dos seus 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 da

- findKey

Error2String

O comando helper Error2String na key_mgmt_util retorna o erro que corresponde a um código de erro hexadecimal da 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, getKeyInfoindica 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

exit

O comando exit em key_mgmt_util sai do 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 da

- Iniciar key_mgmt_util

exportPrivateKey

O comando exportPrivateKey na key_mgmt_util exporta uma chave privada assimétrica em um HSM em 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 da

- importPrivateKey
- wrapKey
- unWrapKey
- genSymKey

exportPubKey

O comando exportPubKey na key_mgmt_util exporta uma chave pública em uma HSM para uma 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 emum 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 da

- importPubKey
- Gerar chaves

exSymKey

O comando exSymKey na ferramenta key_mgmt_util exporta uma cópia em texto simples de uma chave simétrica do HSM e a salva 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 dos seus 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 genSymKeypara 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 imSymKeycomando para importar a chave AES do `aes256-forImport.key` arquivo 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 getKeyInfocomando.

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 empacotamento, use genSymKeypara criar uma chave AES (tipo 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 da

- genSymKey
- imSymKey
- wrapKey

extractMaskedObject

O comando extractMaskedObject na extrai uma chave de um HSM e salva-a 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 da

- insertMaskedObject
- syncKey
- Copiar um backup entre regiões
- Criando um AWS CloudHSM cluster a partir de um backup anterior

findKey

Use o comando findKey findKey key_mgmt_util para procurar chaves usando os valores dos atributos de 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

Estes exemplos mostram como usar findKey para encontrar e identificar chaves nos seus 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, bem como todas as chaves públicas nos 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 getKeyInfoou findAllKeysem 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 IDs de usuário do HSM, como `-u 3` ou `-u 4,7`. Para encontrar os 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 IDs de usuário, 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 de criptografia (CoS) podem usar findAllKeysem 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

Saída

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 da

- findSingleKey
- getKeyInfo
- getAttribute
- findAllKeysem cloudhsm_mgmt_util
- Referência de atributos de chave

findSingleKey

O comando findSingleKey na ferramenta key_mgmt_util verifica se existe uma chave em todos os HSMs no cluster.

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

Este comando verifica se a chave 252136 existe nos 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 da

- findKey
- getKeyInfo
- getAttribute

GendSA KeyPair

O comando genDSAKeyPair na ferramenta key_mgmt_util gera um par de chaves de Algoritmo de assinatura digital (DSA) em seus HSMs. 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 getKeyInfona chave privada (identificador da chave17). 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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 empacotamento 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 de 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 a outros usuários de criptografia (CUs) do HSM permissão para usar essa chave em operações de criptografia. As chaves públicas podem ser usadas por qualquer usuário sem compartilhamento.

Digite uma lista separada por vírgulas de IDs de usuário do HSM, como `-u 5,6`. Não inclua o ID do usuário atual do HSM. Para encontrar o ID do usuário de 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 da

- Gênero A KeyPair
- genSymKey
- GeneCC KeyPair

GeneCC KeyPair

O comando `genECCKeyPair` na ferramenta key_mgmt_util gera um par de chaves de Criptografia de curva elíptica (ECC) nos seus HSMs. 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 o genECCKeyPair para criar pares de chaves ECC nos seus 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 HSMs no modo FIPS, o comando falha. A mensagem informa que um servidor no cluster não está disponível, mas isso geralmente não indica um problema com os 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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 empacotamento 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 de 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 a outros usuários de criptografia (CUs) do HSM permissão para usar essa chave em operações de criptografia. As chaves públicas podem ser usadas por qualquer usuário sem compartilhamento.

Digite uma lista separada por vírgulas de IDs de usuário do HSM, como `-u 5,6`. Não inclua o ID do usuário atual do HSM. Para encontrar o ID do usuário de 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 da

- genSymKey
- Gênero A KeyPair
- GendSA KeyPair

Gênero A KeyPair

O comando genRSAKeyPair na ferramenta key_mgmt_util gera um par de chaves assimétricas 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 nos seus 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 Referência de atributos de chave.

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
d9f27a541676cb9c7def0ef6a2a89c9b9304bcac16fdf8183c0a555421f9ad5dfeb534cf26b65873
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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 o `m_value` de uma 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 empacotamento 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 de 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 a outros usuários de criptografia (CUs) do HSM permissão para usar essa chave em operações de criptografia. As chaves públicas podem ser usadas por qualquer usuário sem compartilhamento.

Digite uma lista separada por vírgulas de IDs de usuário do HSM, como `-u 5,6`. Não inclua o ID do usuário atual do HSM. Para encontrar o ID do usuário de 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 da

- genSymKey
- GendSA KeyPair
- GeneCC KeyPair

genSymKey

O comando genSymKey na ferramenta key_mgmt_util gera uma chave simétrica nos seus HSMs. 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

Estes exemplos mostram como usar o genSymKey para criar chaves simétricas nos seus 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 aguarda a chave a ser sincronizada para todos os 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 for sincronizada com o número mínimo especificado de HSMs antes do término 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, descriptografia 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 Referência de atributos de chave.

```
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. Chaves RC4 não são válidas em HSMs no modo FIPS
- 21: Triple DES (3DES). De acordo com a orientação do NIST, isso não é permitido para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, 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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 o `m_value` de uma 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 empacotamento 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 de 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 a outros usuários de criptografia (CUs) do HSM permissão para usar essa chave em operações de criptografia.

Digite uma lista separada por vírgulas de IDs de usuário do HSM, como `-u 5,6`. Não inclua o ID do usuário atual do HSM. Para encontrar o ID do usuário de 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 da

- exSymKey
- Gênero A KeyPair
- GendSA KeyPair
- GeneCC KeyPair
- setAttribute

getAttribute

O getAttribute comando em key_mgmt_util grava 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 na cloudhsm_mgmt_util, que obtém um valor de atributo de uma chave de todos os 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 Referência de atributos de chave.

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

Estes exemplos mostram como usar getAttribute para obter os atributos de chaves nos seus 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 Referência de atributos de chave.

```
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 Referência de atributos de chave.

```
$  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 Referência de atributos de chave.

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 da

- getAttribute em cloudhsm_mgmt_util
- listAttributes
- setAttribute
- findKey
- Referência de atributos de chave

getCaviumPrivChave

O comando getCaviumPrivKey em key_mgmt_util exporta uma chave privada de um HSM no formato PEM falso. O arquivo PEM falso, que não contém o material da chave privada real, mas, em vez disso, faz referência à chave privada no HSM, pode ser usado para estabelecer o descarregamento de SSL/TLS de seu servidor web para o AWS CloudHSM. Para obter mais informações, consulte Descarregamento de SSL/TLS no Linux.

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 da

- importPrivateKey
- Descarregamento de SSL/TLS no Linux

getCert

O comando getCert na key_mgmt_util recupera certificados de uma partição do HSM e salva-os 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 da

- Verificar a identidade do HSM
- getCert (em cloudhsm_mgmt_util)

getKeyInfo

O comando getKeyInfo na key_mgmt_util retorna os IDs de usuário do HSM dos usuários que podem usar a chave, incluindo o proprietário e os usuários de criptografia (CU) com quem 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 localizar os IDs de usuário do HSM nos seus 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 executa getKeyInfo na chave privada (20), ele retorna o proprietário da chave (3) e os usuários de criptografia (CUs) 4 e 5, com quem 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 GenRSA KeyPair 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 da

- getKeyInfoem cloudhsm_mgmt_util
- listUsers
- findKey
- findAllKeysem cloudhsm_mgmt_util

ajuda

Use o comando help na help da key_mgmt_util para exibir informações sobre todos os comandos disponíveis da key_mgmt_util.

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
insertMaskedObject     Inserts a masked object
extractMaskedObject    Extracts a masked object

     Management Crypto Commands
sign                   Generates a signature
verify                 Verifies a signature
aesWrapUnwrap          Does NIST AES Wrap/Unwrap

     Helper Commands
Error2String           Converts Error codes to Strings
                       save key handle in fake PEM format
getCaviumPrivKey       Saves an RSA private key handle
                       in fake PEM format
IsValidKeyHandlefile   Checks if private key file has
                       an HSM key handle or a real key
listAttributes         List all attributes for getAttributes
listECCCurveIds        List HSM supported ECC CurveIds
```

Parâmetros

Não há parâmetros para esse comando.

Tópicos relacionados da

- loginHSM e logoutHSM

importPrivateKey

O comando importPrivateKey em key_mgmt_util importa uma chave privada assimétrica de um arquivo para um 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.

Example : 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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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

`-timout`

Especifica o número de segundos a aguardar a sincronização da chave entre os HSMs quando o parâmetro `min-serv` 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. Este parâmetro dá a outros usuários de criptografia (CUs) HSM permissão para usar a chave importada em operações criptográficas.

Insira uma lista vírgula separada por vírgulas de IDs de usuários do HSM, como `-u 5,6`. Não inclua o ID do usuário atual do HSM. Para encontrar o ID do usuário de 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 da

- wrapKey
- unWrapKey
- genSymKey
- exportPrivateKey

importPubKey

O comando importPubKey em key_mgmt_util importa uma chave pública de formato PEM para um 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 de HSMs para os quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 a aguardar a sincronização da chave entre os HSMs quando o parâmetro `min-serv` é incluído. Se nenhum número for especificado, a sondagem continuará para sempre.

Padrão: sem limite

Obrigatório: não

Tópicos relacionados da

- exportPubKey
- Gerar chaves

imSymKey

O comando imSymKey na ferramenta key_mgmt_uti importa uma cópia em texto simples de uma chave simétrica de um arquivo no 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 exSymKeycomando 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 unWrapKeyos comandos.

Além disso, o comando imSymKey importa apenas chaves simétricas. Para importar chaves públicas, use importPubKey. Para importar chaves privadas, use importPrivateKeyou 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

Estes exemplos mostram como usar imSymKey para importar chaves simétricas em seus HSMs.

Example : Importar uma chave simétrica AES

Este exemplo usa imSymKey para importar uma chave simétrica AES nos 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-forImport.key 32
```

O segundo comando usa imSymKey para importar a chave AES do arquivo `aes256.key` para os HSMs. 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 exSymKeye imSymKey mover uma chave AES de texto simples entre clusters. Você pode usar um processo como esse para criar um encapsulamento AES que existe HSMs em ambos os clusters. Depois que a chave de empacotamento compartilhada estiver em vigor, você poderá usar WrapKey unWrapKeye mover chaves criptografadas entre os clusters.

O usuário CU que realiza essa operação deve ter permissão para fazer login nos HSMs em ambos os clusters.

O primeiro comando é usado exSymKeypara exportar a chave 14, uma chave AES de 32 bits, do cluster 1 para o `aes.key` arquivo. Ele usa a chave 6, uma chave AES nos HSMs do cluster 1, como a chave de encapsulamento.

```
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"
```

O usuário então faz login em key_mgmt_util no cluster 2 e executa um comando imSymKey para importar a chave no arquivo `aes.key` para os HSMs no cluster 2. Esse comando usa a chave 252152, uma chave AES nos HSMs do cluster 2, como a chave de encapsulamento.

Como as chaves de agrupamento imSymKey usadas agrupam exSymKeye desempacotam imediatamente as chaves de destino, as chaves de agrupamento nos diferentes clusters não precisam ser as mesmas.

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 de chave existe nos HSMs de ambos os clusters, agora você pode compartilhar chaves criptografadas entre eles sem nunca expor a chave em 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 Triple DES com o comprimento especificado. As chaves RC4 e DES não são válidas em HSMs no modo FIPS.

- 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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 empacotamento 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 de 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 a outros usuários de criptografia (CUs) do HSM permissão para usar essa chave em operações de criptografia.

Digite um ID ou uma lista separada por vírgulas de IDs de usuários do HSM, 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 empacotamento, use genSymKeypara criar uma chave AES (tipo 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 da

- genSymKey
- exSymKey
- wrapKey
- unWrapKey

- exportPrivateKey
- exportPubKey

insertMaskedObject

O comando insertMaskedObject em key_mgmt_util insere um objeto mascarado de um arquivo em um 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 descriptografada 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 da

- extractMaskedObject
- syncKey
- Copiar um backup entre regiões
- Criando um AWS CloudHSM cluster a partir de um backup anterior

IsValidKeyHandlefile

O comando IsValidKeyHandlefile em key_mgmt_util é usado 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 descarregamento de SSL/TLS do seu servidor web para o AWS CloudHSM. Para obter mais informações, consulte Descarregamento de SSL/TLS no Linux.

> **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 da

- getCaviumPrivChave
- Descarregamento de SSL/TLS no Linux

listAttributes

O listAttributes comando em key_mgmt_util lista 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 Referência de atributos de chave.

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 Referência de atributos de chave.

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 da

- listAttributes em cloudhsm_mgmt_util

- getAttribute
- setAttribute
- Referência de atributos de chave

listUsers

O listUsers comando em key_mgmt_util obtém os usuários nos HSMs, juntamente com seu tipo de usuário e outros atributos.

Em key_mgmt_util, listUsers retorna a saída que representa todos os HSMs do 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 de HSMs no 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
 LoginFailureCnt         2FA
```

```
    1               1      PCO             admin                   NO
0               NO
    2               2      AU              app_user                NO
0               NO
    3               3      CU              alice                   YES
0               NO
    4               4      CU              bob                     NO
0               NO
    5               5      CU              trent                   YES
0               NO

    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 da

- ListUsers em cloudhsm_mgmt_util
- findKey
- getAttribute
- getKeyInfo

loginHSM e logoutHSM

Os comandos loginHSM e logoutHSM na key_mgmt_util permitem que você faça login e logout dos HSMs em um cluster. Quando conectado aos HSMs, você pode usar a key_mgmt_util para realizar uma variedade de operações de gerenciamento de chaves, incluindo geração, sincronização e encapsulamento de chaves públicas e privadas.

Antes de executar qualquer comando key_mgmt_util, você deve iniciar key_mgmt_util. Para gerenciar chaves com a key_mgmt_util, você deve fazer login nos HSMs como um usuário de criptografia (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 pode ocorrer porque o cliente do CloudHSM divide a carga entre 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 logout de HSMs em um cluster com os comandos `logoutHSM` e `loginHSM`.

Example : Fazer login em HSMs

Esse comando faz login no HSMs como um usuário de criptografia (CU) com o nome de usuário `example_user` e a senha `aws`. A saída mostra que você fez login em todos os 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 aos 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 : Log out of the HSMs

Esse comando faz logout dos HSMs. A saída mostra que você fez logout em todos os 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 da

- exit

setAttribute

O comando setAttribute em key_mgmt_util converte uma chave que é válida apenas na sessão atual em uma chave persistente que existirá até que você a exclua. 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 genSymKeypara 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 Referência de atributos de chave.

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 da

- setAttribute em cloudhsm_mgmt_util
- getAttribute
- listAttributes
- Referência de atributos de chave

sign

O comando sign em key_mgmt_util usa uma chave privada escolhida para gerar uma assinatura de 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

Mecanismo de assinatura	Número inteiro correspondente
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 da

- verify
- importPrivateKey
- Gênero A KeyPair
- GeneCC KeyPair
- genSymKey
- Gerar chaves

unWrapKey

O comando unWrapKey na ferramenta key_mgmt_util importa 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 os HSMs. 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 empacotamento, use genSymKeypara gerar uma chave AES (tipo 31) ou GenRSA KeyPair 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 empacotamento 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 de HSMs nos quais a chave importada é sincronizada antes da expiração do valor do parâmetro `-timeout`. 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 agilizar o processo, configure o valor de `min_srv` como menos 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 de 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
`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 = modulusLengthIn Bytes- (hashLengthIn2* 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 = (Bytes -11). modulusLengthIn

-t

Algoritmo hash	Valor
SHA1	2
SHA256	3
SHA384	4
SHA512	5
SHA224 (válido para RSA_AES e RSA_0AEP 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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Tópicos relacionados da

- wrapKey
- exSymKey
- imSymKey

verificar

O comando verify na key_mgmt_util confirma 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 `hardwarCert.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
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 da

- sign
- getCert
- Gerar chaves

wrapKey

O comando wrapKey na key_mgmt_util exporta uma cópia criptografada de uma chave simétrica ou privada do 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 de volta para o HSM, use unWrapKey. Para exportar uma chave de texto simples de um HSM, use exSymKeyou exportPrivateKeyconforme apropriado. O aesWrapUnwrapcomando 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 Referência de atributos de chave.

Você pode exportar apenas as chaves que você possui. Para encontrar o proprietário de uma chave, use o getKeyInfocomando.

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 genSymKeypara gerar uma chave AES (tipo 31) ou GenRSA KeyPair 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

Mecanismo	Valor
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 = (modulusLengthInBytes-2* Bytes-2). 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 = (Bytes -11). modulusLengthIn

-t

O valor que representa o algoritmo hash. O CloudHSM é compatível com os seguintes algoritmos:

Algoritmo hash	Valor
SHA1	2
SHA256	3

Algoritmo hash	Valor
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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Tópicos relacionados

- exSymKey
- imSymKey
- unWrapKey

Referência de atributos de chave

Os comandos da key_mgmt_util usam constantes para representar os atributos de chaves em um 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 descriptografia, 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 privadapública. 3: Chave privada em um par de chaves privadapública. 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.

Atributo	Constante	Valores
		1: True. A chave pode ser usada para criptografar dados.
OBJ_ATTR_EXTRACTABLE	354	0: False. 1: True. A chave pode ser exportada dos 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.
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 os HSMs. 1: True.

Atributo	Constante	Valores
OBJ_ATTR_MODULUS	288	O módulo que foi usado para gerar um par de chaves RSA. Para chaves EC, este valor representa a codificação DER do valor ANSI X9.62 ECPoint “Q” em formato hexadecimal. Para outros tipos de chave, esse atributo não existe.
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_EXTRACTABLE	356	0: False. 1: True. Não é possível exportar a chave dos 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.

Atributo	Constante	Valores
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.
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.

Atributo	Constante	Valores
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.
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 encapsulada usando essa chave de encapsulamento.
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.

AWS CloudHSM SDKs do cliente

Use um Client SDK para descarregar 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 as suporte de plataforma, consulte Plataformas compatíveis com o Client SDK 5.

Para obter mais informações sobre como usar Client SDK 3, consulte SDK do cliente anterior (SDK do cliente 3).

the section called “Biblioteca PKCS #11”

O PKCS #11 é um padrão para realizar operações de criptografia em módulos de segurança de hardware (HSMs). AWS CloudHSM oferece implementações da biblioteca PKCS #11 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 JCE”

O provedor AWS CloudHSM JCE é compatível com a Arquitetura Criptográfica Java (JCA). O provedor permite que você execute operações criptográficas no HSM.

the section called “Provedores de KSP e CNG”

O AWS CloudHSM cliente para Windows inclui provedores de CNG e KSP. Atualmente, somente o Client SDK 3 é compatível com provedores de CNG e KSP.

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 de plataforma para um determinado componente, primeiro

certifique-se de que a plataforma desejada apareça na seção base do SDK e, em seguida, verifique se há exclusões ou outras informações pertinentes 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 ter mais informações, consulte Downloads para o SDK AWS CloudHSM do cliente.

Para ver as plataformas compatíveis com o Client SDK anterior, consulte Plataformas compatíveis com o Client SDK 3

O Client SDK 5 não exige um daemon de cliente.

Suporte Linux para o Client SDK 5

Plataformas compatíveis	Arquitetura X86_64	Arquitetura ARM
Amazon Linux 2	Sim	Sim
Amazon Linux 2023	Sim	Sim
CentOS 7 (7.8+)	Sim	Não
Red Hat Enterprise Linux 7 (7.8+)	Sim	Não
Red Hat Enterprise Linux 8 (8.3+)	Sim	Não
Red Hat Enterprise Linux 9 (9.2+)	Sim	Sim
Ubuntu 20.04 LTS	Sim	Não
Ubuntu 22.04 LTS	Sim	Sim

Nota: 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 Client SDK 5

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Suporte de tecnologia sem servidor para o Client SDK 5

- AWS Lambda
- Docker/ECS

Compatibilidade com HSM para o Client SDK 5

hsm1.medium	hsm2m.medium
Compatível com o SDK do cliente versão 5.0.0 e posterior.	Compatível com o SDK do cliente versão 5.12.0 e posterior.

Suporte a componentes

CLI do CloudHSM

A CLI do CloudHSM é uma ferramenta de linha de comando que ajuda os administradores a gerenciar usuários em seu cluster. Para ter mais informações, consulte Interface de linha de comando (CLI) do CloudHSM.

Biblioteca PKCS #11

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 Linux para o Client SDK 5” e the section called “Suporte do Windows para o Client SDK 5”.

Mecanismo dinâmico do OpenSSL

O OpenSSL Dynamic Engine é um componente exclusivo do Linux que requer o OpenSSL 1.0.2, 1.1.1 ou 3.x.

Provedor JCE

O provedor JCE é um SDK Java compatível com OpenJDK 8, OpenJDK 11, OpenJDK 17 e OpenJDK 21 em todas as plataformas suportadas.

Benefícios do Client SDK 5

Comparado ao 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.

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.

Migração do Client SDK 3 para o Client SDK 5

Para obter instruções detalhadas sobre a migração do SDK do cliente 3 para o SDK do cliente 5, consulte as instruções de migração para cada SDK do cliente individual:

- Migre sua biblioteca PKCS #11 do Client SDK 3 para o Client SDK 5
- Migre seu OpenSSL Dynamic Engine do Client SDK 3 para o Client SDK 5
- Migre seu provedor de JCE do Client SDK 3 para o Client SDK 5
- Migrar do SDK 3 do cliente (CMU e KMU) para o SDK do cliente 5 (CloudHSM CLI)

Para funcionalidades ou casos de uso que não são compatíveis com a CLI do CloudHSM, entre em contato com o suporte.

> **Note**
>
> A biblioteca PKCS #11 do Client SDK 5 agora é suportada em plataformas Windows. Ele pode lidar com a maioria dos casos de uso em que os fornecedores de CNG e KSP podem e devem ser considerados substitutos. Atualmente, o KSP só está disponível no Client SDK 3.

Biblioteca PKCS #11

O PKCS #11 é um padrão para realizar operações de criptografia em módulos de segurança de hardware (HSMs). AWS CloudHSM oferece implementações da biblioteca PKCS #11 compatíveis com a versão 2.40 do PKCS #11.

Para obter informações sobre bootstrapping, consulte Conectar-se ao cluster. Para solução de problemas, consulteProblemas conhecidos da biblioteca do PKCS#11.

Para obter mais informações sobre como usar Client SDK 3, consulte SDK do cliente anterior (SDK do cliente 3).

Tópicos

Instale a biblioteca PKCS #11 para o Client SDK 5

Este tópico fornece instruções para instalar a versão mais recente da biblioteca PKCS #11 para git, a série Client SDK 5. Para obter mais informações sobre a biblioteca Client SDK ou do PKCS #11, consulte Usando a biblioteca Client SDK e PKCS #11.

Instalação

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 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 arquitetura ARM64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-pkcs11-latest.el7.aarch64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-pkcs11-latest.el7.aarch64.rpm
   ```

   Amazon Linux 2023

   Instale a biblioteca PKCS #11 para 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 arquitetura ARM64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/cloudhsm-pkcs11-latest.amzn2023.aarch64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-pkcs11-latest.amzn2023.aarch64.rpm
   ```

CentOS 7 (7.8+)

Instale a biblioteca PKCS #11 para CentOS 7.8+ 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
```

RHEL 7 (7.8+)

Instale a biblioteca PKCS #11 para RHEL 7 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
```

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
```

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 arquitetura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-pkcs11-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el9.aarch64.rpm
```

Ubuntu 20.04 LTS

Instale a biblioteca PKCS #11 para Ubuntu 20.04 LTS na arquitetura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-pkcs11_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u20.04_amd64.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 2016

Instale a biblioteca PKCS #11 para Windows Server 2016 na arquitetura X86_64:

1. Faça o download da biblioteca PKCS #11 para Client SDK 5.

2. Execute o instalador da biblioteca PKCS #11 (AWSCloudHSMPKCS11-latest.msi) com privilégios administrativos do Windows.

Windows Server 2019

Instale a biblioteca PKCS #11 para Windows Server 2019 na arquitetura X86_64:

1. Faça o download da biblioteca PKCS #11 para Client SDK 5.
2. Execute o instalador da biblioteca PKCS #11 (AWSCloudHSMPKCS11-latest.msi) com privilégios administrativos do Windows.

2. Use a ferramenta de configuração para especificar um local para o certificado de emissão. Para obter 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:\ProgramFiles\Amazon\CloudHSM
     ```

     Scripts de configuração e arquivos de log do Windows:

     ```
     C:\ProgramData\Amazon\CloudHSM
     ```

Autentique-se na biblioteca PKCS #11

Quando você usar a biblioteca PKCS #11, seu aplicativo será executado como um usuário de criptografia (CU) específico em seus HSMs. O aplicativo pode visualizar e gerenciar apenas as chaves que o CU possui e compartilha. Você pode usar um CU existente em seus HSMs ou criar um novo CU para seu aplicativo. Para obter informações sobre como gerenciar CUs, consulte Gerenciar usuários HSM com a CLI do CloudHSM e Gerenciar usuários HSM com 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

A biblioteca de PKCS #11 suporta os tipos de chave a seguir.

Tipo de chave	Descrição
AES	Gere chaves AES de 128, 192 e 256 bits.
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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Mecanismos suportados para a 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.

A biblioteca de software do para PKCS #11 oferece suporte aos seguintes algoritmos:

- Criptografia e descriptografia: AES-CBC, AES-CTR, AES-ECB, AES-GCM, DES3-CBC, DES3-ECB, RSA-OAEP e RSA-PKCS
- Assinar e verificar: RSA, HMAC e ECDSA; com e sem hash
- Hash/Digest: SHA1, SHA224, SHA256, SHA384 e SHA512
- Encapsulamento de chave: AES Key Wrap[1], AES-GCM, RSA-AES e RSA-OAEP

Tópicos

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_HMAC[2]
- CKM_SHA224_HMAC[2]
- CKM_SHA256_HMAC[2]
- CKM_SHA384_HMAC[2]
- CKM_SHA512_HMAC[2]

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 da biblioteca PKCS #11 permite que você use os seguintes mecanismos para funções do Derive.

- CKM_SP800_108_COUNTER_KDF

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_GCM[3]
- CKM_RSA_AES_KEY_WRAP
- CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD[3]
- CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD[3]
- CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD[3]

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

Mecanismo	Tamanho máximo de dados em bytes
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/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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Operações de API suportadas para a biblioteca PKCS #11

A biblioteca de software PKCS #11 oferece suporte às operações de API do PKCS #11 a seguir.

- `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-chave suportados pela biblioteca PKCS #11

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 o SDK do PKCS #11 do CloudHSM, atribuímos valores padrão conforme especificado pelo PKCS #11 padrão.

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.

Interpretar a tabela de atributos do PKCS #11

A tabela do PKCS #11 contém uma lista de atributos que diferem por tipos de chaves. 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
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	Verdadeiro

Atributo	Tipo de chave				Valor padrão
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	✓	✖	✓	✖	
CKA_SENSITIVE	✓[1]	✖	✓[1]	✖	Verdadeiro

Atributo	Tipo de chave				Valor padrão
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	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	

Atributo	Tipo de chave				Valor padrão
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	✓	✓	✓	

Atributo	Tipo de chave			Valor padrão
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
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	Verdadeiro
CKA_VERIFY_RECOVER	✖	✖	✖	
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	✖	✖	✖	
CKA_EXTRACTABLE	✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_MODULUS	✖	✖	✖	
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	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	
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_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
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	Falso
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	Verdadeiro

Atributo	Tipo de chave							Valor padrão
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
CKA_UNWRAP_TEMPLATE	✓	✗	✓	✗	✓	✓	✗	
CKA_SENSITIVE	✓	✗	✓	✗	✓	✓	✓	Verdadeiro

Atributo	Tipo de chave							Valor padrão
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	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✓	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✓[2]	✖	✖	✖	✖	

Atributo	Tipo de chave							Valor padrão
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	✓	✓	✓	✓	✓	

Atributo	Tipo de chave					Valor padrão
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
CKA_DESTROYABLE	✓	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✓	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✖	✖	✖	✖	✖	Falso
CKA_VERIFY	✖	✖	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	

Atributo	Tipo de chave					Valor padrão
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	✖	✖	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	✖	✖	

Atributo	Tipo de chave					Valor padrão
CKA_EXPONENT_2	✖	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	✖	
CKA_VALUE	✖	✖	✖	✖	✖	
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]	

Atributo	Tipo de chave			Valor padrão
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
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	Verdadeiro
CKA_DESTROYABLE	✓[1]	✓[1]	✓[1]	Verdadeiro
CKA_SIGN	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	Falso

Atributo	Tipo de chave			Valor padrão
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	
CKA_MODULUS	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	

Atributo	Tipo de chave			Valor padrão
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]	
CKA_CHECK_VALUE	R	R	R	

GetAttributeValue

Atributo	Tipo de chave						
	EC privada	EC pública	RSA privada	RSA pública	AES	DES3	Segredo genérico

Atributo	Tipo de chave						
CKA_CLASS	✓	✓	✓	✓	✓	✓	✓
CKA_KEY_TYPE	✓	✓	✓	✓	✓	✓	✓
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓
CKA_ID	✓	✓	✓	✓	✓	✓	✓
CKA_LOCAL	✓	✓	✓	✓	✓	✓	✓
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]
CKA_ENCRYPT	✖	✖	✖	✓	✓	✓	✖
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓

Atributo	Tipo de chave						
CKA_SIGN_RECOVER	✖	✖	✓	✖	✖	✖	✖
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✖	✖	✖	✓	✖	✖	✖
CKA_WRAP	✖	✖	✖	✓	✓	✓	✖
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✓	✓	✖
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓

Atributo	Tipo de chave						
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	✖	✖	✖	✖
CKA_EXPONENT_2	✖	✖	S	✖	✖	✖	✖
CKA_PRIVATE_EXPONENT	✖	✖	S	✖	✖	✖	✖
CKA_PUBLIC_EXPONENT	✖	✖	✓	✓	✖	✖	✖

Atributo	Tipo de chave						
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.

Modificar atributos

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.

Interpretar códigos de erro

Especificar no modelo um atributo 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 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 conformid ade com a especificação do PKCS #11, mas não tem suporte do CloudHSM.
`CKR_ATTRIBUTE_INCOMPLETE`	Você receberá esse erro quando não especific ar o atributo obrigatório no modelo de atributo.
`CKR_ATTRIBUTE_READ_ONLY`	Você receberá esse erro quando especific ar um atributo somente leitura no modelo de atributo.

Exemplos de código para a biblioteca PKCS #11

Os exemplos de código mostrados GitHub mostram como realizar tarefas básicas usando a biblioteca PKCS #11.

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.

- Generate keys (AES, RSA, EC)
- List key attributes
- Criptografar e descriptografar dados com AES-GCM
- Encrypt and decrypt data with AES_CTR
- Encrypt and decrypt data with 3DES
- Sign and verify data with 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

Migre sua biblioteca PKCS #11 do Client SDK 3 para o Client SDK 5

Use este tópico para migrar sua biblioteca PKCS #11 do SDK do cliente 3 para o SDK do cliente 5. Para obter os benefícios da migração, consulteBenefícios do 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 continua a ter novos recursos e suporte de plataforma adicionados a ele.

Para revisar as instruções de migração para todos os provedores, consulteMigração do Client SDK 3 para o Client SDK 5.

Prepare-se abordando as mudanças mais importantes

Revise essas alterações importantes e atualize seu aplicativo em seu ambiente de desenvolvimento adequadamente.

Os mecanismos de embalagem foram alterados

Mecanismo Client SDK 3	Mecanismo equivalente do SDK 5 do cliente
`CKM_AES_KEY_WRAP`	`CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD`
`CKM_AES_KEY_WRAP_PAD`	`CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD`
`CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD`	`CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD`
`CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD`	`CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD`
`CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD`	`CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_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 seu aplicativo precisar dessa funcionalidade, entre em contato com o suporte.

Os identificadores de teclas 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 aplicativos existentes que usarão os mesmos identificadores de chave em sessões diferentes, você deverá modificar seu código para obter o identificador de chave sempre que executar o aplicativo. 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 SDK do cliente 3 para o SDK do cliente 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 o Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

2. Desinstale o daemon do cliente para o SDK do cliente 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
```

> **Note**
>
> As configurações personalizadas precisam ser ativadas novamente.

3. Instale a biblioteca Client SDK PKCS #11 seguindo as etapas em. Instale a biblioteca PKCS #11 para o Client SDK 5
4. 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 abaixo. Bootstrap o Client SDK
5. Em seu ambiente de desenvolvimento, teste seu aplicativo. Faça atualizações em seu código existente para resolver suas alterações importantes antes da migração final.

Tópicos relacionados

- Práticas recomendadas para AWS CloudHSM

Configurações avançadas para PKCS #11

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

Conectando-se a vários slots com PKCS#11

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, é possível configurar sua biblioteca PKCS11 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

- 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 configurar sua biblioteca PKCS #11 para a funcionalidade de vários slots, 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.

configure-pkcs11 add-cluster

Ao se conectar a vários slots com o PKCS #11, use o comando configure-pkcs11 add-cluster 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>]
        [--server-client-cert-file <CLIENT CERTIFICATE FILE>]
        [--server-client-key-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 de`cluster-1234567`) para a sua configuração.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 add-cluster --cluster-id cluster-1234567
```

Windows

```
C:\Program Files\Amazon\CloudHSM\> .\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

```
        C:\Program Files\Amazon\CloudHSM\> .\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

--cluster-id ***<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

--endpoint ***<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 do arquivo para o certificado CA do HSM.

Obrigatório: Não

--region ***<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

-- server-client-cert-file <Client Certificate Filepath>

Caminho para o certificado de cliente usado para autenticação mútua entre cliente e servidor TLS.

Use essa opção somente se não quiser usar a chave padrão e o certificado SSL/TLS que incluímos no Client SDK 5. Você deve definir essa opção em combinação com `--server-client-key-file`.

Obrigatório: Não

-- server-client-key-file <Client Key Filepath>

Caminho para a chave do cliente usada para autenticação mútua entre cliente e servidor TLS.

Use essa opção somente se não quiser usar a chave padrão e o certificado SSL/TLS que incluímos no Client SDK 5. Você deve definir essa opção em combinação com `--server-client-cert-file`.

Obrigatório: Não

configure-pkcs11 remove-cluster

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

```
C:\Program Files\Amazon\CloudHSM\> .\configure-pkcs11.exe remove-cluster --cluster-id cluster-1234567
```

Para obter mais informações sobre o parâmetro `--cluster-id`, consulte the section called “Parâmetros”.

Parâmetro

--cluster-id *<Cluster ID>*

O ID do cluster a ser removido da configuração

Obrigatório: Sim

Repetir comandos para PKCS #11

O Client SDK 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações com controle de utilização pelo HSM do lado do cliente. Quando um HSM controla a utilização das operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, os SDKs do cliente tentarão repetir as operações com controle de utilização até 3 vezes enquanto recuam 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, os SDKs do cliente vão repetir automaticamente as operações com controle de utilização recuando exponencialmente.

Para ter 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:

```
C:\Program Files\Amazon\CloudHSM\bin\ .\configure-pkcs11.exe --default-retry-
mode off
```

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.

AWS CloudHSM fornece um mecanismo dinâmico OpenSSL, sobre o qual você pode ler. Descarregamento de SSL/TLS no Linux Para ver um exemplo de uso AWS CloudHSM com o OpenSSL, consulte este blog de segurança da AWS. Para obter mais informações sobre o suporte de plataforma para SDKs, consulte the section called “Plataformas compatíveis”. Para solução de problemas, consulteProblemas conhecidos para o OpenSSL Dynamic Engine.

Para obter mais informações sobre como usar Client SDK 3, consulte SDK do cliente anterior (SDK do cliente 3).

Para obter mais informações, consulte os tópicos abaixo.

Tópicos

- Instalação do Mecanismo dinâmico do OpenSSL
- Tipos de chaves do OpenSSL Dynamic Engine

- Mecanismos do OpenSSL Dynamic Engine
- Migre seu OpenSSL Dynamic Engine do Client SDK 3 para o Client SDK 5
- Configurações avançadas para OpenSSL

Instalação do Mecanismo dinâmico do OpenSSL

> **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 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
   ```

   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
```

CentOS 7 (7.8+)

Instale o OpenSSL Dynamic Engine para CentOS 7 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
```

RHEL 7 (7.8+)

Instale o OpenSSL Dynamic Engine para RHEL 7 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
```

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
```

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
```

Ubuntu 20.04 LTS

Instale o OpenSSL Dynamic Engine para Ubuntu 20.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-dyn_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u20.04_amd64.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 de usuários de criptografia (CUs), consulte Usar o CMU para gerenciar usuários.

```
$ 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`.

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 ter 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 chaves do OpenSSL Dynamic Engine

O AWS CloudHSM OpenSSL Dynamic Engine suporta os seguintes tipos de chave.

Tipo de chave	Descrição
EC	Assine/verifique ECDSA para os tipos de chaves P-256, P-384 e secp256k1. Para gerar chaves EC que sejam interoperáveis com o mecanismo OpenSSL, consulte key generate-file.
RSA	Geração de chaves RSA para chaves de 2048, 3072 e 4096 bits. Assina/verificação RSA. A verificação é descarregada para o software OpenSSL.

Mecanismos do OpenSSL Dynamic Engine

Saiba como usar os mecanismos do AWS CloudHSM OpenSSL Dynamic Engine.

Funções de assinatura e verificação

O AWS CloudHSM OpenSSL Dynamic Engine 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.

Tipos de assinatura RSA

- SHA1withRSA
- SHA224withRSA
- SHA256withRSA

- SHA384withRSA
- SHA512withRSA

Tipos de assinatura ECDSA

- SHA1withECDSA
- SHA224withECDSA
- SHA256withECDSA
- SHA384withECDSA
- SHA512withECDSA

Migre seu OpenSSL Dynamic Engine do Client SDK 3 para o Client SDK 5

Use este tópico para migrar seu OpenSSL Dynamic Engine do SDK do cliente 3 para o SDK do cliente 5. Para obter os benefícios da migração, consulteBenefícios do 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 continua a ter novos recursos e suporte de plataforma adicionados a ele.

> **Note**
>
> Atualmente, a geração de números aleatórios não é suportada no Client SDK 5 com OpenSSL Dynamic Engine.

Para revisar as instruções de migração para todos os provedores, consulteMigração do Client SDK 3 para o Client SDK 5.

Migrar para o Client SDK 5

Siga as instruções nesta seção para migrar do SDK do cliente 3 para o SDK do cliente 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 for Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

2. Desinstale o daemon do cliente para o SDK do cliente 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
```

> **Note**
>
> As configurações personalizadas precisam ser ativadas novamente.

3. Instale o Client SDK OpenSSL Dynamic Engine seguindo as etapas em. Instalação do Mecanismo dinâmico do OpenSSL
4. O Client SDK 5 apresenta um novo formato de arquivo de configuração e uma ferramenta de inicialização de linha de comando. Para inicializar seu Client SDK 5 OpenSSL Dynamic Engine, siga as instruções listadas no guia do usuário abaixo. Bootstrap o Client SDK
5. Em seu ambiente de desenvolvimento, teste seu aplicativo. Faça atualizações em seu código existente para resolver suas alterações importantes antes da migração final.

Tópicos relacionados da

- Práticas recomendadas para AWS CloudHSM

Configurações avançadas para OpenSSL

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

Repetir comandos para OpenSSL

O Client SDK 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações com controle de utilização pelo HSM do lado do cliente. Quando um HSM controla a utilização das operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, os SDKs do cliente tentarão repetir as operações com

controle de utilização até 3 vezes enquanto recuam 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, os SDKs do cliente vão repetir automaticamente as operações com controle de utilização recuando exponencialmente.

Para ter 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

- 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
```

Windows

Para definir comandos de repetição off para o Client SDK 5 no Windows

- Você pode usar os comandos a seguir para definir os comandos de repetição para o modo off:

```
C:\Program Files\Amazon\CloudHSM\bin\ .\configure-dyn.exe --default-retry-mode
 off
```

Provedor JCE

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, consulteProblemas conhecidos do SDK do JCE.

Para obter mais informações sobre como usar Client SDK 3, consulte SDK do cliente anterior (SDK do cliente 3).

Tópicos

- Instale e use o provedor AWS CloudHSM JCE para o Client SDK 5
- Tipos de chaves compatíveis com o provedor de JCE
- Mecanismos suportados pelo provedor de JCE
- Atributos de chave Java compatíveis
- Exemplos de código para a biblioteca AWS CloudHSM de software para Java
- AWS CloudHSM Javadocs, provedor de JCE
- Usando a classe AWS CloudHSM KeyStore Java
- Migre seu provedor de JCE do Client SDK 3 para o Client SDK 5
- Configurações avançadas para JCE

Instale e use o provedor AWS CloudHSM JCE para o Client SDK 5

O provedor JCE é compatível com OpenJDK 8, OpenJDK 11, OpenJDK 17 e OpenJDK 21. Você pode baixar os dois no site do OpenJDK.

> **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

- Instalar o provedor JCE
- Forneça credenciais ao provedor JCE
- Fundamentos do gerenciamento de chaves no provedor JCE

Instalar o provedor JCE

1. Use so seguintes comandos para fazer download e instalar o provedor JCE.

   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 arquitetura ARM64:

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

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

   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 arquitetura ARM64:

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

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

CentOS 7 (7.8+)

Instale o provedor JCE para CentOS 7 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
```

RHEL 7 (7.8+)

Instale o provedor JCE para RHEL 7 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
```

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
```

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
```

Ubuntu 20.04 LTS

Instale o provedor JCE para o Ubuntu 20.04 LTS na arquitetura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-jce_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u20.04_amd64.deb
```

Ubuntu 22.04 LTS

Instale o provedor JCE para o 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 2016

Instale o provedor JCE para 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/AWSCloudHSMJCE-latest.msi -Outfile C:\AWSCloudHSMJCE-latest.msi
```

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

Windows Server 2019

Instale o provedor JCE para 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/AWSCloudHSMJCE-latest.msi -Outfile C:\AWSCloudHSMJCE-latest.msi
```

```
PS C:\> Start-Process msiexec.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`

Forneça credenciais ao provedor JCE

Os HSMs precisam autenticar seu aplicativo Java antes que o aplicativo possa usá-los. Os HSMs autenticam uma sessão usando o 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 mome 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.

  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

Fundamentos do gerenciamento de chaves no provedor JCE

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.

Tipos de chaves compatíveis com o provedor de JCE

A biblioteca de AWS CloudHSM software para Java permite gerar os seguintes tipos de chaves.

Tipo de chave	Descrição
AES	Gere chaves AES de 128, 192 e 256 bits.
DES triplo (3DES, DESEDE)	Gere uma chave DES tripla de 192 bits. Veja a nota de rodapé 1 para ver uma mudança futura.
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.

Tipo de chave	Descrição
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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Mecanismos suportados pelo provedor de JCE

Este tópico fornece informações sobre os mecanismos suportados pelo provedor JCE com o 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)
- 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`.
AES	GCM	`AES/GCM/NoPadding`	Implementa `Cipher.WRAP_MODE`, `Cipher.UNWRAP_MODE`

Algoritmo	Modo	Padding	Observações
			, `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` .

DESede (Triple DES)	CBC	`DESede/CBC/PKCS5Padding`
		
		`DESede/CBC/NoPadding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` . Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações

DESede (Triple DES)	ECB	`DESede/ECB/NoPadding`
		
		`DESede/ECB/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`. Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações

RSA	ECB	`RSA/ECB/PKCS1Padding` consulte a observação 1
		
		`RSA/ECB/OAEPPadding`
		
		`RSA/ECB/OAEPWithSHA-1ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-224ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-256ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-384ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-512ANDMGF1Padding`
			Implementa `Cipher.WRAP_MODE`, `Cipher.UNWRAP_MODE`, `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`.

Algoritmo	Modo	Padding	Observações
RSA	ECB	`RSA/ECB/NoPadding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`.

RSAAESWrap	ECB	`RSAAESWrap/ECB/OAEPPadding`
		
		`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 oferece suporte aos 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)

Os CMACs (códigos de autenticação de mensagens baseados em cifras) criam códigos de autenticação de mensagens (MACs) usando uma cifra de bloqueio e uma chave secreta. Eles diferem dos HMACs porque usam um método de chave simétrica de bloco para os MACs em vez de um método de hashing.

A biblioteca AWS CloudHSM de software para Java suporta os seguintes algoritmos CMAC.

- `AESCMAC`

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 KeySpecclasses SecretKeyFactory do `generateSecret` método For são suportadas:
  - KeyAttributesMappode ser usado para importar bytes de chave com atributos adicionais como uma chave do CloudHSM. Um exemplo pode ser encontrado aqui
  - SecretKeySpecpode ser usado para importar uma especificação de chave simétrica como uma chave do CloudHSM.
  - AesCmacKdfParameterSpecpode ser usado para derivar chaves simétricas usando outra chave AES do CloudHSM.

> **Note**
>
> SecretKeyFactoryO `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 suportada KeySpec para importar uma chave assimétrica. AWS CloudHSM Para obter mais informações, consulte os seguintes recursos:

- Para KeyFactory o `generatePublic` método de For, KeySpecas 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

  - Chave pública RSA PublicKeySpec para RSA
  - EC PublicKeySpec para chave pública EC
- Para KeyFactory o `generatePrivate` método de For, KeySpecas 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
  - PKCS8 EncodedKeySpec para chave privada EC e RSA
  - Chave privada RSA PrivateCrtKeySpec para RSA
  - EC PrivateKeySpec para chave privada EC

KeyFactoryO `translateKey` método de For, ele usa qualquer chave que implemente a interface chave.

Anotações do mecanismo

[1] De acordo com a orientação do NIST, isso não é permitido para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Atributos de chave Java compatíveis

Este tópico fornece informações sobre os atributos de chave Java compatíveis com o 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 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 púbicas, 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 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.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
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 para identificar a chave.
KEY_TYPE				Usado para identificar o tipo de chave (AES, DeSede, genérica secreta, EC ou RSA).

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
LABEL				Uma string definida pelo usuário que permite que você identifiq ue convenien temente 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.
OBJECT_CL ASS				Usado para identificar a classe de objeto de uma chave (SecretKey PublicKey ou PrivateKey).

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
PRIVATE	TRUE	TRUE	TRUE	True indica que um usuário pode não acessar a chave até que o usuário seja autentica do. Para maior clareza, os usuários não podem acessar nenhuma chave AWS CloudHSM até serem autenticados, mesmo que esse atributo esteja definido como FALSE.
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.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
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.
TOKEN	FALSE	FALSE	FALSE	Uma chave permanente que é replicada em todos os HSMs no cluster e incluída em backups. TOKEN = FALSE implica uma chave efêmera que é apagada automaticamente quando a conexão com o HSM é interrompida ou é feito logout.

Atributo	Valor padrão			Observações
	Chave simétrica	Chave pública no par de chaves	Chave privada no par de chaves	
UNWRAP	TRUE		TRUE	True indica que você pode usar a chave para desencapsular (importar) outra chave.
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 do **`KeyAttributesMap`**
Obter o valor de um atributo de chave para uma chave existente	Objeto (contendo o valor) ou nulo	get(keyAttribute)
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	put(keyAttribute, valor)
Preencher valores para múltiplos atributos de chave	N/D	putAll () keyAttributesMap
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	remove(keyAttribute)

> **Note**
>
> Todos os atributos que você não especificar explicitamente serão definidos como os padrões listados na tabela anterior em the section called “Atributos 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

Este tópico fornece recursos e informações sobre exemplos de código Java para o Client SDK 5.

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 decrpyt with AES-CTR
- Criptografar e descriptografar com DESede-ECBver consulte a nota 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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, 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 Versão mais recente seção na página Downloads.

Você pode importar Javadocs em um ambiente de desenvolvimento integrado (IDE) ou visualizá-los em um navegador da Web.

Usando a classe AWS CloudHSM KeyStore Java

A AWS CloudHSM `KeyStore` classe fornece um armazenamento de chaves PKCS12 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 uso`KeyStore`, 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 do repositório de chaves apropriado

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")
```

Inicializando AWS CloudHSM KeyStore

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 arquivo PKCS12 local que contém dados de repositório de chaves. 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);
```

Usando AWS CloudHSM KeyStore

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 arquivo PKCS12 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 repositório de chaves verifica objetos armazenados localmente no arquivo PKCS12 e objetos residentes no HSM.

- `deleteEntry`

  Exclui uma entrada de certificado do arquivo PKCS12 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 Destrutível.

```
((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 procura o alias no arquivo PKCS12 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 repositório de chaves no fluxo de saída fornecido como arquivo PKCS12 e protege-o com a senha fornecida. Além disso, mantém todas as chaves carregadas (que são definidas usando chamadas `setKey`).

Migre seu provedor de JCE do Client SDK 3 para o Client SDK 5

Use este tópico para migrar seu provedor de JCE do SDK do cliente 3 para o SDK do cliente 5. Para obter os benefícios da migração, consulteBenefícios do 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 continua a ter novos recursos e suporte de plataforma adicionados a ele.

O provedor Client SDK 3 JCE usa classes e APIs personalizadas que não fazem parte da especificação JCE padrão. O SDK 5 do cliente para o provedor JCE está em conformidade com a especificação JCE e é incompatível com versões anteriores do SDK do cliente 3 em determinadas áreas. Os aplicativos 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, consulteMigração do Client SDK 3 para o Client SDK 5.

Tópicos

- Prepare-se abordando as mudanças mais importantes
- Migrar para o Client SDK 5
- Tópicos relacionados

Prepare-se abordando as mudanças mais importantes

Analise essas alterações importantes e atualize seu aplicativo 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 `CaviumProvider` e tem o nome do provedor. `Cavium`	No Client SDK 5, a classe Provider é chamada `CloudHsmProvider` e tem o nome `CloudHSM` Provider.	Um exemplo de como inicializar o `CloudHsmProvider` 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 SDK 3 do cliente usa a `LoginManager` classe para login explícito. [1]	No Client SDK 5, o `CloudHSM` provedor implementa o login `AuthProvider` explícito. `AuthProvider` é uma classe Java padrão e segue a forma idiomática a do Java de fazer login em um provedor. Com o gerenciamento aprimorado do estado de login no Client SDK 5, os aplicativos não precisam mais monitorar e realizar o login durante as [2] reconexões.	Para ver um exemplo de como usar o login explícito com o SDK 5 do cliente, consulte o LoginRunner exemplo no repositório de amostras do AWS GitHub CloudHSM.
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 SDK do cliente 3 para o SDK do cliente 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 SDK 3 do cliente:

```
LoginManager lm = LoginManager.getInstance();
```

```
lm.login(partition, user, pass);
```

- [2] Trecho de código do SDK 5 do cliente:

```
// 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, `Cavium[Key-type]AlgorithmParameterSpec` 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, `KeyAttributesMap` 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 `KeyAttributesMap` para gerar uma chave simétrica, consulte a SymmetricKeys amostra no repositório de amostras do AWS CloudHSM Github.
Geração de pares de chaves	No Client SDK 3, `Cavium[Key-type]Algorithmparameterspec` é usado para especific	No Client SDK 5, `KeyPairAttributesMap` é usado para especificar esses parâmetros. Para obter um trecho	Para ver um exemplo de como usar `KeyAttributesMap` para gerar uma chave assimétrica, consulte

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
	ar parâmetros de geração de pares de chaves. Para obter um trecho de código, consulte a nota de rodapé. 3	de código, consulte a nota de rodapé. 4	a AsymmetricKeys amostra no repositór io de AWS CloudHSM GitHub amostras.

- [1] Trecho de código de geração de chave 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 chave 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 5 pares de chaves do SDK do cliente:

```
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 referenciar chaves, enquanto o Client SDK 5 usa rótulos de chave. As mudanças de comportamento resultantes estão listadas abaixo.

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Referências principais	Com o Client SDK 3, os aplicativos usam rótulos ou identific adores de teclas para referenciar chaves no HSM. Eles usam rótulos com KeyStore para encontrar uma chave ou usam alças para criar `CaviumKey` objetos.	No Client SDK 5, os aplicativos podem usar o Usando a classe AWS CloudHSM KeyStore Java para encontrar chaves por rótulo. Para encontrar as chaves pela alça, use o AWS CloudHSM `KeyStoreW ithAttributes` com AWS CloudHSM `KeyRefereneSpec` .	
Encontrando várias entradas	Ao pesquisar uma chave usando`getEntry`,`getK` ou `getCertif icate` em cenários em que existem vários itens com os mesmos critérios no `Cavium` KeyStore, somente a primeira	Com o AWS CloudHSM `KeyStore` `eKeyStoreW ithAttrib utes` , esse mesmo cenário resultará no lançamento de uma exceção. Para corrigir esse problema, é recomendável definir	

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
	entrada encontrada será retornada.	rótulos exclusivos para chaves usando o atributo do conjunto de chaves comando na CLI do CloudHSM. Ou use `KeyStoreWithAttributes#getKeys` para retornar todas as chaves que correspondam aos critérios.	

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Encontre todas as chaves	É possível no Client SDK 3 encontrar todas as chaves no HSM usando. `Util.findAllKeys()`	O SDK 5 do cliente torna a localizaç ão de chaves mais simples e eficiente usando a `KeyStoreWithAttributes` classe. Quando possível, armazene suas chaves em cache para minimizar a latência. Para ter mais informações, consulte Gerencie com eficácia as chaves em seu aplicativo. Quando precisar recuperar todas as chaves do HSM, use `KeyStoreWithAttributes#getKeys` com um vazio. `KeyAttributesMap`	Um exemplo que usa a `KeyStoreWithAttributes` classe para encontrar uma chave está disponível no repositório de amostra do AWS CloudHSM Github e um trecho de código é mostrado em. 1

O que mudou	O que estava no Client SDK 3	O que está no Client SDK 5	Exemplo
Exclusão de chaves	O SDK 3 do cliente usa `Util.deleteKey()` para excluir uma chave.	O `Key` objeto no Client SDK 5 implementa a `Destroyable` interface que permite que as chaves sejam excluídas usando o `destroy()` método dessa interface.	Um exemplo de código mostrando a funcionalidade de exclusão da chave pode ser encontrado no repositório de amostra do Github do 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] Excluindo uma chave no SDK do cliente 3:

```
Util.deleteKey(key);
```

  Excluindo uma chave no Client SDK 5:

```
((Destroyable) key).destroy();
```

As operações de desempacotamento de cifras foram alteradas, outras operações de cifragem não

> **Note**
>
> Nenhuma alteração é necessária nas operações de criptografia/descriptografia/empacotamento do Cipher.

As operações de desempacotamento exigem que a `CaviumUnwrapParameterSpec` classe Client SDK 3 seja substituída por uma das seguintes classes específicas das operações criptográficas listadas.

- `GCMUnwrapKeySpec`para `AES/GCM/NoPadding` desembrulhar
- `IvUnwrapKeySpec`para `AESWrap unwrap` e `AES/CBC/NoPadding unwrap`
- `OAEPUnwrapKeySpec` para `RSA OAEP unwrap`

Exemplo de trecho 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 nas operações de assinatura.

Migrar para o Client SDK 5

Siga as instruções nesta seção para migrar do SDK do cliente 3 para o SDK do cliente 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. Desinstale o provedor JCE para o Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-jce
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-jce
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-jce
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-jce
   ```

2. Desinstale o daemon do cliente para o SDK do cliente 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
```

> **Note**
>
> As configurações personalizadas precisam ser ativadas novamente.

3. Instale o provedor JCE do SDK do cliente seguindo as etapas em. Instale e use o provedor AWS CloudHSM JCE para o Client SDK 5
4. 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 do Client SDK 5 JCE, siga as instruções listadas no guia do usuário abaixo. Bootstrap o Client SDK
5. Em seu ambiente de desenvolvimento, teste seu aplicativo. Faça atualizações em seu código existente para resolver suas alterações importantes antes da migração final.

Tópicos relacionados

- Práticas recomendadas para AWS CloudHSM

Configurações avançadas para JCE

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 clusters com o provedor JCE

Essa configuração permite que uma única instância cliente se comunique com vários clusters. 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());
}
```

Configuração de **`CloudHsmProvider`**

`CloudHsmProvider`pode ser configurado de duas maneiras:

1. Configurar com arquivo (configuração padrão)
2. Configurar usando código

Configurar com arquivo (configuração padrão)

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 usando código

A partir do Client SDK versão 5.8.0, você também pode configurar o `CloudHsmProvider` 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 do JCE é`CloudHsmCluster1`. Esse é o nome que o aplicativo 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.

Conectando-se a vários clusters

Conforme mencionado acima, cada `CloudHsmProvider` representa uma conexão com seu cluster do CloudHSM. 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 operação NoPadding AES/GCM/:

```
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());
```

Repetir comandos para o JCE

O Client SDK 5.8.0 e versões posteriores têm uma estratégia de repetição automática integrada que repetirá as operações com controle de utilização pelo HSM do lado do cliente. Quando um HSM controla a utilização das operações porque está muito ocupado executando operações anteriores e não pode receber mais solicitações, os SDKs do cliente tentarão repetir as operações com controle de utilização até 3 vezes enquanto recuam 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, os SDKs do cliente vão repetir automaticamente as operações com controle de utilização recuando exponencialmente.

Para ter 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:

```
C:\Program Files\Amazon\CloudHSM\bin\ .\configure-jce.exe --default-retry-mode
 off
```

Extração de chaves usando JCE

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 da chave dos seus HSMs em texto não criptografado na memória da sua máquina para uso. Normalmente, os HSMs só permitem que as chaves sejam extraídas como objetos agrupados, não como texto não criptografado. 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

Classe	Método	Format (getEncoded)
	getW()	N/D
RSA PublicKey	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.

Permitindo que o provedor de AWS CloudHSM JCE extraia segredos de chave privada do AWS CloudHSM

> **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

   ```
   C:\Program Files\Amazon\CloudHSM\> .\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
CE PrivateKey	getEncoded()	PKCS#8
	getS()	N/D
RSA PrivateCrtKey	getEncoded()	X.509
	getPrivateExponent()	N/D
	getPrimeP()	N/D
	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

```
C:\Program Files\Amazon\CloudHSM\> .\configure-jce.exe --disable-clear-key-extraction-in-software
```

Cryptography API: Next Generation (CNG – API de criptografia da próxima geração) e provedor de armazenamento de chaves (KSP) para Microsoft Windows

O AWS CloudHSM cliente para Windows inclui provedores de CNG e KSP. Atualmente, somente o Client SDK 3 é compatível com provedores de CNG e KSP.

Os Key storage providers [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 ter 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: permitem 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

Instalar os provedores de KSP e CNG para Windows

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).

Configurar e executar o cliente AWS CloudHSM no 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 precisará executar essas etapas na primeira vez 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 todos os 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

Esta etapa usa o parâmetro -a da Configure tool (ferramenta Configure) para adicionar o endereço IP da interface de rede elástica (ENI) de um dos HSMs do cluster ao arquivo de configuração.

```
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 DescribeClustersoperação, 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 os HSMs do 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.

```
C:\Program Files\Amazon\CloudHSM>ksp_config.exe -enum
```

```
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.

```
C:\Program Files\Amazon\CloudHSM>cng_config.exe -register
```

Se o provedor KSP estiver ausente, execute o comando a seguir.

```
C:\Program Files\Amazon\CloudHSM>ksp_config.exe -register
```

AWS CloudHSM Pré-requisitos do Windows

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 localizar os CUs existentes, 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 do 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 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 -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 ProviderPrivate 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 o provedor Cavium CNG

⚠ ** Somente código de exemplo – não para 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 AWS CloudHSM Pré-requisitos do Windows.

```
// 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;
}
```

SDK do cliente anterior (SDK do cliente 3)

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 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 a migração do SDK do cliente 3 para o SDK do cliente 5, consulte. Migração do Client SDK 3 para o Client SDK 5

A documentação do Client SDK 3 está listada neste tópico.

Para fazer download do Downloads.

Verifique a versão do SDK do seu cliente

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
```

Comparação de componentes do Client SDK

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 SDK 5 do cliente tem paridade com o SDK do cliente 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
Provedores de KSP e CNG		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

Componente	Client SDK 5	Client SDK 3
Configure a ferramenta	Sim	Sim

[1] Os componentes CMU e KMU estão incluídos na CLI do CloudHSM com o Client SDK 5.

Tópicos

Plataformas compatíveis com o Client SDK 3

Client SDK 3 requer um daemon de cliente e oferece ferramentas de linha de comando, incluindo a CloudHSM Management Utility (CMU), a 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 de plataforma para um determinado componente, primeiro certifique-se de que a plataforma desejada apareça na seção base do SDK e, em seguida, verifique se há exclusões ou outras informações pertinentes 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 ter mais informações, consulte Downloads para o SDK AWS CloudHSM do cliente.

AWS CloudHSM suporta somente sistemas operacionais de 64 bits.

Sumário

- Suporte a componentes
  - Biblioteca PKCS #11
  - CloudHSM Management Utility (CMU – utilitário de gerenciamento do CloudHSM)
  - Key Management Utility (KMU – utilitário de gerenciamento de chaves)
  - Provedor JCE
  - Mecanismo dinâmico do OpenSSL
  - Provedores de KSP e CNG

Suporte a 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 ao Windows

- Microsoft Windows Server 2012
- Microsoft Windows Server 2012 R2

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Compatibilidade com HSM para Client SDK 3

hsm1.medium	hsm2m.medium
Compatível com a versão Client SDK 3.1.0 e posterior.	Sem suporte.

Suporte a componentes

Biblioteca PKCS #11

A biblioteca PKCS #11 é um componente exclusivo do Linux que corresponde ao suporte básico do Linux. Para ter mais informações, consulte the section called “Suporte a Linux”.

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 nos HSMs. Inclui ferramentas que criam, excluem e listam usuários e alteram senhas de usuários. Para ter mais informações, consulte CloudHSM Management Utility (CMU – utilitário de gerenciamento do CloudHSM).

Key Management Utility (KMU – utilitário de gerenciamento de chaves)

O Key Management Utility (KMU) é uma ferramenta de linha de comando que ajuda usuários criptográficos (UC) a gerenciar chaves nos módulos de segurança de hardware (HSM). Para ter mais informações, consulte Key Management Utility (KMU – utilitário de gerenciamento de chaves).

Provedor JCE

O provedor JCE é um componente exclusivo do Linux que corresponde ao suporte básico do Linux. Para ter mais informações, consulte the section called “Suporte a Linux”.

- Requer o OpenJDK 1.8

Mecanismo dinâmico do OpenSSL

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. AWS CloudHSM suporta essas plataformas com o OpenSSL Dynamic Engine para Client SDK 5.

Provedores de KSP e CNG

Os provedores de CNG e KSP são um componente exclusivo do Windows que corresponde ao suporte básico do Windows. Para ter mais informações, consulte Suporte ao Windows.

Atualizar o Client SDK 3 no Linux

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

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-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 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
```

Ubuntu 20.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 22.04 LTS

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

Biblioteca PKCS #11 para Client SDK 3

O PKCS #11 é um padrão para realizar operações criptográficas em hardware security modules (HSM, módulos de segurança de hardware).

Para obter informações sobre bootstrapping, consulte Conectar-se ao cluster.

Tópicos

- Instalando a biblioteca PKCS #11 para o Client SDK 3
- Autenticação na biblioteca PKCS #11 (Client SDK 3)
- Tipos de chaves compatíveis (Client SDK 3)
- Mecanismos suportados (Client SDK 3)
- Operações de API suportadas (Client SDK 3)
- Atributos de chave suportados (Client SDK 3)
- Exemplos de código para a biblioteca PKCS #11 (Client SDK 3)

Instalando a biblioteca PKCS #11 para o Client SDK 3

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 emInstalar 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 cloudhsm-client start
```

CentOS 7

```
$ sudo systemctl cloudhsm-client start
```

CentOS 8

```
$ sudo systemctl cloudhsm-client start
```

RHEL 7

```
$ sudo systemctl cloudhsm-client start
```

RHEL 8

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 20.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

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
```

Autenticação na biblioteca PKCS #11 (Client SDK 3)

Quando você usar a biblioteca PKCS #11, seu aplicativo será executado como um usuário de criptografia (CU) específico em seus HSMs. O aplicativo pode visualizar e gerenciar apenas as chaves que o CU possui e compartilha. Você pode usar um CU existente em seus HSMs ou criar um novo CU. Para obter informações sobre como gerenciar CUs, consulte Gerenciar usuários HSM com a CLI do CloudHSM e Gerenciar usuários HSM com 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 (Client SDK 3)

A biblioteca de PKCS #11 suporta os tipos de chave a seguir.

Tipo de chave	Descrição
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.

Tipo de chave	Descrição
DES3 (Triple DES)	Gera chaves DES3 de 192 bits. 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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Mecanismos suportados (Client SDK 3)

A biblioteca de software do para PKCS #11 oferece suporte aos seguintes algoritmos:

- Criptografia e descriptografia: AES-CBC, AES-CTR, AES-ECB, AES-GCM, DES3-CBC, DES3-ECB, RSA-OAEP e RSA-PKCS
- Assinar e verificar: RSA, HMAC e ECDSA; com e sem hash
- Hash/Digest: SHA1, SHA224, SHA256, SHA384 e SHA512
- Encapsulamento de chave: AES Key Wrap[4], 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	Criptografar e descriptografar	Derivar chave	Embrulhe e UnWrap
CKM_RSA_PKCS_KEY_PAIR_GEN	✓						
CKM_RSA_X9_31_KEY_PAIR_GEN	✓[2]						
CKM_RSA_X_509		✓			✓		
CKM_RSA_PKCS consulte a observação 8		✓[1]	✖		✓[1]		✓[1]
CKM_RSA_PKCS_OAEP					✓[1]		✓[6]
CKM_SHA1_RSA_PKCS		✓[3.2]					
CKM_SHA224_RSA_PKCS		✓[3.2]					

Mecanismo	Funções						
CKM_SHA256_RSA_PKCS		✓[3.2]					
CKM_SHA384_RSA_PKCS		✓[2,3.2]					
CKM_SHA512_RSA_PKCS		✓[3.2]					
CKM_RSA_PKCS_PSS		✓[1]					
CKM_SHA1_RSA_PKCS_PSS		✓[3.2]					
CKM_SHA224_RSA_PKCS_PSS		✓[3.2]					
CKM_SHA256_RSA_PKCS_PSS		✓[3.2]					
CKM_SHA384_RSA_PKCS_PSS		✓[2,3.2]					
CKM_SHA512_RSA_PKCS_PSS		✓[3.2]					

Mecanismo	Funções						
CKM_EC_KEY_PAIR_GEN	✓						
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]	
CKM_SP800_108_COUNTER_KDF						✓	
CKM_GENERIC_SECRET_KEY_GEN	✓						
CKM_AES_KEY_GEN	✓						

Mecanismo	Funções						
CKM_AES_ECB					✓		✖
CKM_AES_CTR					✓		✖
CKM_AES_CBC					✓[3.3]		✖
CKM_AES_CBC_PAD					✓		✖
CKM_DES3_KEY_GEN consulte a observação 8	✓						
CKM_DES3_CBC consulte a observação 8					✓[3.3]		✖
CKM_DES3_CBC_PAD consulte a observação 8					✓		✖

Mecanismo	Funções						
CKM_DES3_ECB consulte a observação 8					✓		✖
CKM_AES_GCM					✓[3.3, 4]		✓[7.1]
CKM_CLOUDHSM_AES_GCM					✓[7.1]		✓[7.1]
CKM_SHA_1				✓[3.1]			
CKM_SHA_1_HMAC		✓[3.3]					
CKM_SHA224				✓[3.1]			
CKM_SHA224_HMAC		✓[3.3]					
CKM_SHA256				✓[3.1]			
CKM_SHA256_HMAC		✓[3.3]					
CKM_SHA384				✓[3.1]			
CKM_SHA384_HMAC		✓[3.3]					

Mecanismo	Funções						
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_PKCS5_PAD							✓[7.1]
CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_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
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
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 do PKCS#11. `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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Operações de API suportadas (Client SDK 3)

A biblioteca de software PKCS #11 oferece suporte às operações de API do PKCS #11 a seguir.

- 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 de chave suportados (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 o SDK do PKCS #11 do CloudHSM, atribuímos valores padrão conforme especificado pelo PKCS #11 padrão.

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.

Interpretar a tabela de atributos do PKCS #11

A tabela do PKCS #11 contém uma lista de atributos que diferem por tipos de chaves. 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`	✓	✓	✓	✓	

Atributo	Tipo de chave				Valor padrão
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
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✖	✓	✖	Falso
CKA_SIGN_RECOVER	✖	✖	✓[3]	✖	
CKA_VERIFY	✖	✓	✖	✓	Falso
CKA_VERIFY_RECOVER	✖	✖	✖	✓[4]	

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	R	✖	R	✖	
CKA_EXTRACTABLE	✓	✖	✓	✖	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	✖	R	✖	
CKA_MODULUS	✖	✖	✖	✖	

Atributo	Tipo de chave				Valor padrão
CKA_MODULUS_BITS	✖	✖	✖	✓[2]	
CKA_PRIME_1	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✓[2]	
CKA_EC_PARAMS	✖	✓[2]	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	
CKA_VALUE	✖	✖	✖	✖	

Atributo	Tipo de chave				Valor padrão
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

Atributo	Tipo de chave			Valor padrão
CKA_DERIVE	✓	✓	✓	Falso
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✓	✓	Verdadeiro
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

Atributo	Tipo de chave			Valor padrão
CKA_UNWRAP_TEMPLATE	✓	✓	✖	
CKA_SENSITIVE	✓	✓	✓	Verdadeiro
CKA_ALWAYS_SENSITIVE	✖	✖	✖	
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	✖	✖	✖	

Atributo	Tipo de chave			Valor padrão
CKA_EXPONENT_2	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	
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]	

Atributo	Tipo de chave							Valor padrão
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
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	Falso
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	Verdadeiro
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✖	✖	✓[3]	✖	✖	✖	✖	Falso

Atributo	Tipo de chave							Valor padrão
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✖	✖	✖	✓[4]	✖	✖	✖	
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	R	✖	R	✖	R	R	R	
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓	Verdadeiro

Atributo	Tipo de chave							Valor padrão
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	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✓	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✓[2]	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✓[2]	✓[2]	✖	✖	✖	
CKA_EC_PARAMS	✓[2]	✓[2]	✖	✖	✖	✖	✖	

Atributo	Tipo de chave							Valor padrão
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

Atributo	Tipo de chave					Valor padrão
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
CKA_DESTROYABLE	✓	✓	✓	✓	✓	Verdadeiro
CKA_SIGN	✓	✓	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✖	✓[3]	✖	✖	✖	Falso
CKA_VERIFY	✖	✖	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	
CKA_WRAP	✖	✖	✓	✓	✖	Falso
CKA_UNWRAP	✖	✓	✓	✓	✖	Falso

Atributo	Tipo de chave					Valor padrão
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	✖	✖	✖	✖	✖	
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	✖	✖	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	✖	
CKA_VALUE	✖	✖	✖	✖	✖	
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]	

Atributo	Tipo de chave			Valor padrão
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	✓[1]	✓[1]	✓[1]	Verdadeiro
CKA_SIGN	✓	✓	✓	Falso
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	Falso
CKA_VERIFY_RECOVER	✖	✖	✖	

Atributo	Tipo de chave			Valor padrão
CKA_WRAP	✓	✓	✖	Falso
CKA_UNWRAP	✓	✓	✖	Falso
CKA_SENSITIVE	✓	✓	✓	Verdadeiro
CKA_EXTRACTABLE	✓	✓	✓	Verdadeiro
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	
CKA_MODULUS	✖	✖	✖	
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	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	
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	✓	✓	✓	✓	✓	✓	✓

Atributo	Tipo de chave						
CKA_KEY_TYPE	✓	✓	✓	✓	✓	✓	✓
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓
CKA_ID	✓	✓	✓	✓	✓	✓	✓
CKA_LOCAL	✓	✓	✓	✓	✓	✓	✓
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]
CKA_ENCRYPT	✖	✖	✖	✓	✓	✓	✖
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓
CKA_SIGN_RECOVER	✖	✖	✓	✖	✖	✖	✖

Atributo	Tipo de chave						
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✖	✖	✖	✓	✖	✖	✖
CKA_WRAP	✖	✖	✖	✓	✓	✓	✖
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✓	✓	✖
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓

Atributo	Tipo de chave						
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	✖	✖	✖	✖
CKA_EXPONENT_2	✖	✖	S	✖	✖	✖	✖
CKA_PRIVATE_EXPONENT	✖	✖	S	✖	✖	✖	✖
CKA_PUBLIC_EXPONENT	✖	✖	✓	✓	✖	✖	✖
CKA_EC_PARAMS	✓	✓	✖	✖	✖	✖	✖

Atributo	Tipo de chave						
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.

Modificar atributos

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.

Interpretar códigos de erro

Especificar no modelo um atributo 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 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 conformid ade com a especificação do PKCS #11, mas não tem suporte do CloudHSM.
`CKR_ATTRIBUTE_INCOMPLETE`	Você receberá esse erro quando não especific ar o atributo obrigatório no modelo de atributo.
`CKR_ATTRIBUTE_READ_ONLY`	Você receberá esse erro quando especific ar um atributo somente leitura no modelo de atributo.

Exemplos de código para a biblioteca PKCS #11 (Client SDK 3)

Os exemplos de código mostrados GitHub mostram como realizar tarefas básicas usando a biblioteca PKCS #11.

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.

- Generate keys (AES, RSA, EC)
- List key attributes
- Criptografar e descriptografar dados com AES-GCM
- Encrypt and decrypt data with AES_CTR
- Encrypt and decrypt data with 3DES
- Sign and verify data with 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

Instalação do Client SDK 3 para o mecanismo dinâmico do OpenSSL

O Client SDK 3 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.

Tópicos

Pré-requisitos para o OpenSSL Dynamic Engine com Client SDK 3

Para obter mais informações sobre as suporte de plataforma, consulte Plataformas compatíveis com o Client SDK 3.

Antes de usar o mecanismo AWS CloudHSM dinâmico para OpenSSL, você precisa do cliente. AWS CloudHSM

O cliente é um daemon que estabelece comunicação end-to-end criptografada com os HSMs em seu cluster, e o mecanismo do OpenSSL se comunica localmente com o cliente. Para instalar e configurar o AWS CloudHSM cliente, consulteInstalar o cliente (Linux). Use o comando a seguir para iniciar.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl cloudhsm-client start
```

CentOS 6

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl cloudhsm-client start
```

RHEL 6

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Instalação e utilização do mecanismo dinâmico OpenSSL para o Client SDK 3

As etapas a seguir descrevem como instalar e configurar o mecanismo AWS CloudHSM dinâmico para OpenSSL. Para informações sobre a atualização, consulte Atualizar o 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>
```

Usar o mecanismo dinâmico do OpenSSL 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
```

Client SDK 3 para provedor JCE

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 e use o provedor AWS CloudHSM JCE para o Client SDK 3
- Mecanismos suportados para o Client SDK 3
- Atributos de chave suportados para o Client SDK 3
- Exemplos de código para a biblioteca AWS CloudHSM de software para Java for Client SDK 3
- Usando a classe AWS CloudHSM KeyStore Java para o Client SDK 3

Instale e use o provedor AWS CloudHSM JCE para o 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 os HSMs no 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 emInstalar 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 cloudhsm-client start
```

CentOS 7

```
$ sudo systemctl cloudhsm-client start
```

CentOS 8

```
$ sudo systemctl cloudhsm-client start
```

RHEL 7

```
$ sudo systemctl cloudhsm-client start
```

RHEL 8

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 20.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Tópicos

Instalar o provedor JCE

Use so 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 Atualizar o 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`

Validação da 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 *<nome do usuário do HSM>* e *<senha>* pelas credenciais de um usuário de criptografia (CU).

```
$ 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)
```

Fornecimento de credenciais ao provedor JCE

Os HSMs precisam autenticar seu aplicativo Java antes que o aplicativo possa usá-los. Cada aplicativo pode usar uma sessão. Os HSMs autenticam uma sessão usando o método de login explícito ou 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 mome 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 torna o tratamento de erros difícil de compreender quando as credenciais são inválidas ou os HSMs estiverem com problemas na autenticação da sessão.

Fundamentos do gerenciamento de chaves no provedor JCE

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 suportados para o 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
- Mecanismos de assinatura/verificação compatíveis
- 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/NoPadding` `AES/CBC/PKCS5Padding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .

Algoritmo	Modo	Padding	Observações

AES	ECB	`AES/ECB/NoPadding`
		
		`AES/ECB/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`. Use a transformação de AES.

AES	CTR	`AES/CTR/NoPadding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_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 (Triple DES)	CBC	`DESede/CBC/NoPadding`
		
		`DESede/CBC/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .
			
			As rotinas de geração de chaves aceitam um tamanho de 168 ou 192 bits. No entanto, internamente, todas as chaves DESede são de 192 bits.
			
			Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações

DESede (Triple DES)	ECB	`DESede/ECB/NoPadding`
		
		`DESede/ECB/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`.
			
			As rotinas de geração de chaves aceitam um tamanho de 168 ou 192 bits. No entanto, internamente, todas as chaves DESede são de 192 bits.
			
			Consulte a nota 1 abaixo para ver uma mudança futura.

RSA	ECB	`RSA/ECB/NoPadding`
		
		`RSA/ECB/PKCS1Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`.
			
			Consulte a nota 1 abaixo para ver uma mudança futura.

Algoritmo	Modo	Padding	Observações

RSA	ECB	`RSA/ECB/OAEPPadding`
		
		`RSA/ECB/OAEPWithSHA-1ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-224ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-256ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-384ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-512ANDMGF1Padding`
			Implementa `Cipher.ENCRYPT_MODE`, `Cipher.DECRYPT_MODE`, `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`.
			
			`OAEPPadding` é OAEP com o tipo de preenchimento `SHA-1`.

RSAAESWrap	ECB	`OAEPPADDING`	Implementa `Cipher.WRAP_Mode` e `Cipher.UNWRAP_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`

Mecanismos de assinatura/verificação compatíveis

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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Atributos de chave suportados para o Client SDK 3

Este tópico descreve como usar uma extensão proprietária para a biblioteca Java versão 3.1 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
- 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

Use atributos para especificar quais ações são permitidas em objetos de chave, incluindo chaves púbicas, 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 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 em todos os HSMs no cluster e incluída em backups. CKA_TOKEN = FALSE requer uma chave de sessão, que é carregada somente em um HSM e apagada automatic amente quando a conexão com o HSM é interromp ida.
CKA_LABEL				Uma string definida pelo usuário. Ela permite identific ar chaves no HSM de forma conveniente.
CKA_EXTRA CTABLE	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 . Geralmente, 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 autentica do. 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 do **CloudHSMKeyAttributesMap**
Obter o valor de um atributo de chave para uma chave existente	Objeto (contendo o valor) ou nulo	get(keyAttribute)
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	put(keyAttribute, valor)
Preencher valores para múltiplos atributos de chave	N/D	putAll () keyAttributesMap
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	remove(keyAttribute)

> **Note**
>
> Todos os atributos que você não especificar explicitamente serão definidos como os padrões listados na tabela anterior em the section called “Atributos 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	`CaviumKeyGenAlgorithmParameterSpec`	`CaviumKeyGenAlgorithmParameterSpec(CloudHsmKeyAttributesMap keyAttributesMap)`
DES	`CaviumDESKeyGenParameterSpec`	`CaviumDESKeyGenParameterSpec(int keySize, byte[] iv, CloudHsmKeyAttributesMap keyAttributesMap)`

Tipo de chave	Classe de especificação de parâmetros	Exemplo de construtores
RSA	`CaviumRSAKeyGenParameterSpec`	`CaviumRSAKeyGenParameterSpec(int keysize, BigInteger publicExponent, CloudHsmKeyPairAttributesMap keyPairAttributesMap)`
Secreta	`CaviumGenericSecretKeyGenParameterSpec`	`CaviumGenericSecretKeyGenParameterSpec(int size, CloudHsmKeyAttributesMap keyAttributesMap)`
AES	`CaviumAESKeyGenParameterSpec`	`CaviumAESKeyGenParameterSpec(int keySize, byte[] iv, CloudHsmKeyAttributesMap keyAttributesMap)`
EC	`CaviumECGenParameterSpec`	`CaviumECGenParameterSpec(String stdName, CloudHsmKeyPairAttributesMap keyPairAttributesMap)`

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

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 decrpyt 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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, ainda é permitido após 2023. Para mais detalhes, consulte Conformidade com o FIPS 140: suspensão do mecanismo de 2024.

Usando a classe AWS CloudHSM KeyStore Java para o Client SDK 3

A AWS CloudHSM **`KeyStore`** classe fornece um armazenamento de chaves PKCS12 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 uso`KeyStore`, consulte Classe KeyStore.

Escolha do repositório de chaves apropriado

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")
```

Inicializando AWS CloudHSM KeyStore

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 arquivo PKCS12 local que contém dados de repositório de chaves. 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);
```

Usando AWS CloudHSM KeyStore

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 arquivo PKCS12 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 repositório de chaves verifica objetos armazenados localmente no arquivo PKCS12 e objetos residentes no HSM.

- `DeleteEntry`

  Exclui uma entrada de certificado do arquivo PKCS12 local. A exclusão de dados importantes 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 procura o alias no arquivo PKCS12 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 repositório de chaves no fluxo de saída fornecido como arquivo PKCS12 e protege-o com a senha fornecida. Além disso, mantém todas as chaves carregadas (que são definidas usando chamadas `setKey`).

Integrar 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 descarregamento de SSL/TLS em 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 pública SSL/TLS 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. Descarregar reduz a carga computacional dos servidores Web e oferece segurança extra ao armazenar as chaves privadas do servidor em HSMs.

Os tópicos a seguir fornecem uma visão geral de como o descarregamento de SSL/TLS AWS CloudHSM funciona e tutoriais para configurar o descarregamento de SSL/TLS 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

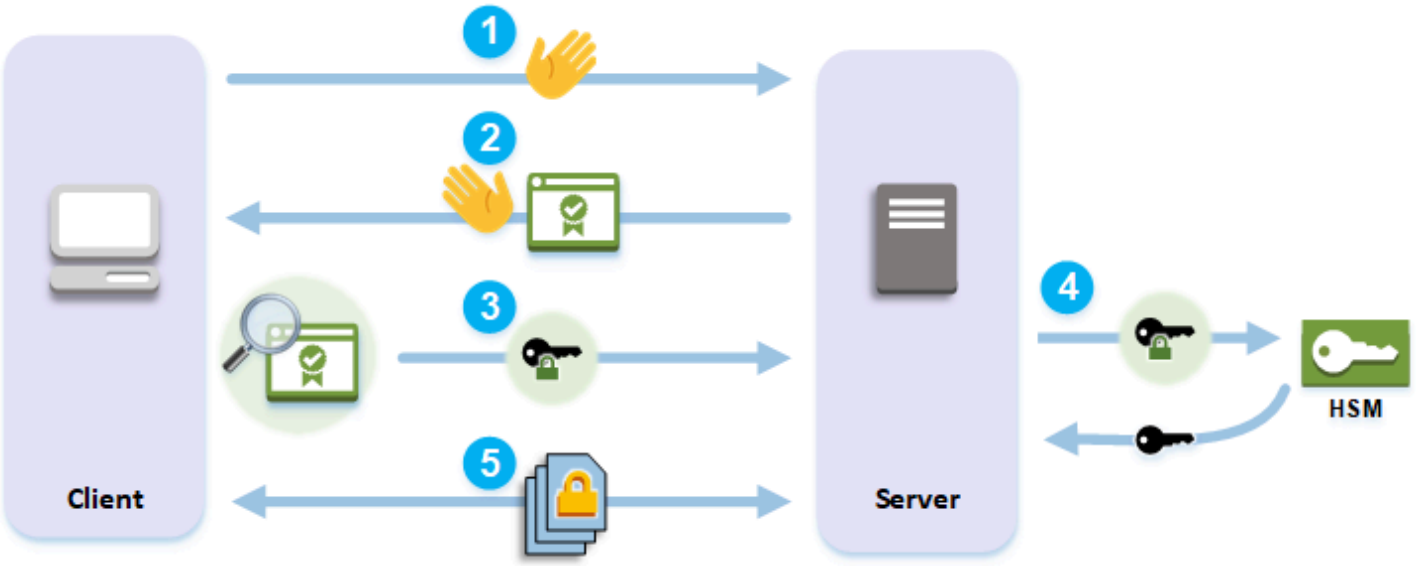
Tópicos

- Como funciona o descarregamento de SSL/TLS AWS CloudHSM
- Descarregamento de SSL/TLS no Linux
- Usar IIS com CNG para descarregamento de SSL/TLS no Windows
- Adicione um balanceador de carga com o Elastic Load Balancing (opcional)

Como funciona o descarregamento de SSL/TLS AWS CloudHSM

Para estabelecer uma conexão HTTPS, seu servidor Web executa um processo de handshake com os clientes. Como parte desse processo, o servidor descarrega parte do processamento criptográfico nos HSMs, conforme mostrado 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 do servidor SSL/TLS está assinado por um certificado raiz em que o cliente confia.
   b. Extrai a chave pública do certificado do servidor.
   c. Gera um segredo pré-mestre 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 descarregamento de SSL/TLS com AWS CloudHSM, consulte um dos tópicos a seguir:

- Descarregamento de SSL/TLS no Linux
- Usar IIS com CNG para descarregamento de SSL/TLS no Windows

Descarregamento de SSL/TLS no Linux

Com AWS CloudHSM, você pode realizar o descarregamento de SSL/TLS no Linux com NGINX, Apache e Tomcat. Para obter mais informações, consulte os tópicos abaixo.

Tópicos

- Usando NGINX ou Apache com OpenSSL para descarregamento de SSL/TLS no Linux
- Usando o Tomcat com JSSE para descarregamento de SSL/TLS no Linux

Usando NGINX ou Apache com OpenSSL para descarregamento de SSL/TLS no Linux

Este tópico fornece step-by-step instruções para configurar o descarregamento de SSL/TLS AWS CloudHSM em um servidor web Linux.

Tópicos

Visão geral

No Linux, o software de servidor Web NGINX e o Apache HTTP Server são integrados ao OpenSSL para comportar HTTPS. O mecanismo dinâmico do AWS CloudHSM para OpenSSL fornece uma interface que permite que o software de servidor web use os HSMs em seu cluster para descarregamento criptográfico 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.

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 descarregamento SSL/TLS de servidor Web com o Client SDK 5, é necessário o 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
```

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

3. Use a CLI do CloudHSM para criar um 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.

Depois de concluir essas etapas, vá para Etapa 2: Gerar ou importar uma chave privada e um certificado SSL/TLS.

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 ter mais informações, consulte Adicionar um HSM.
- 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.

Pré-requisitos para o Client SDK 3

Para configurar o descarregamento SSL/TLS de servidor Web com o Client SDK 3, é necessário o seguinte:

- Um AWS CloudHSM cluster ativo com pelo menos um HSM.
- Uma instância do Amazon EC2 executando um sistema operacional Linux com o seguinte software instalado:
  - O AWS CloudHSM cliente e as ferramentas da linha de comando.
  - O aplicativo do servidor web NGINX ou Apache.
  - O mecanismo AWS CloudHSM dinâmico do OpenSSL.
- 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. Siga as etapas em Conceitos básicos. Em seguida, você terá um cluster ativo com um HSM e uma instância de cliente do Amazon EC2. Sua instância do EC2 será configurada com as ferramentas da linha de comando. Use essa instância de cliente como seu servidor Web.

2. Conecte-se 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.
3. 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

   - A versão 1.19 do NGINX é a versão mais recente do NGINX compatível com o mecanismo SDK do Client SDK 3 no Amazon Linux 2.

     Para obter mais informações e baixar a versão 1.19 do NGINX, consulte o site do NGINX.

   - Apache

     ```
     $ sudo yum install httpd mod_ssl
     ```

   CentOS 7

   - A versão 1.19 do NGINX é a versão mais recente do NGINX compatível com o mecanismo do Client SDK 3 no CentOS 7.

     Para obter mais informações e baixar a versão 1.19 do NGINX, consulte o site do NGINX.

   - Apache

     ```
     $ sudo yum install httpd mod_ssl
     ```

Red Hat 7

- A versão 1.19 do NGINX é a versão mais recente do NGINX compatível com o mecanismo Client SDK 3 no Red Hat 7.

  Para obter mais informações e baixar a versão 1.19 do NGINX, consulte o site do NGINX.

- Apache

```
$ sudo yum install httpd mod_ssl
```

Ubuntu 16.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 18.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

4. (Opcional) Adicione mais HSMs ao seu cluster. Para ter mais informações, consulte Adicionar um HSM.
5. Use cloudhsm_mgmt_util para criar um CU. Para ter mais informações, consulte Gerenciamento de usuários de HSM. 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 um certificado SSL/TLS.

Etapa 2: Gerar ou importar uma chave privada e um certificado SSL/TLS

Para habilitar o HTTPS, o aplicativo do servidor Web (NGINX ou Apache) precisa de uma chave privada e de um certificado SSL/TLS correspondente. Para usar o servidor web SSL/TLS offload com AWS CloudHSM, você deve armazenar a chave privada em um HSM no seu cluster. AWS CloudHSM Você pode conseguir isso de uma das seguintes maneiras:

- Se você ainda não tem uma chave privada e um certificado correspondente, gere uma chave privada em um HSM. Em seguida, use a chave privada para criar uma solicitação de assinatura de certificado (CSR), que é então assinada para produzir um certificadoSSL/TLS.
- Se você já tiver uma chave privada e um certificado correspondente, importe a chave privada para um HSM.

Independentemente de qual dos métodos anteriores você escolher, 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 da chave privada PEM falsa para identificar a chave privada no HSM durante o descarregamento de SSL/TLS.

Execute um destes procedimentos:

- Gerar uma chave privada e um certificado
- Importar uma chave privada e um certificado existentes

Gerar uma chave privada e um certificado

Gerar uma chave privada

Esta seção mostra como gerar um par de chaves usando o Key Management Utility (KMU) do Client SDK 3. Depois de gerar um par de chaves dentro do HSM, você pode exportá-lo como um arquivo PEM falso e gerar o certificado correspondente.

As chaves privadas geradas com o Key Management Utility (KMU) podem ser usadas com o Client SDK 3 e Client SDK 5.

Instalar e configurar o Key Management Utility (KMU)

1. Conecte-se à instância do cliente.
2. Instale e configure o Client SDK 3.
3. Execute 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
   ```

Ubuntu 20.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 22.04 LTS

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

4. Execute o comando a seguir para iniciar a ferramenta da linha de comando key_mgmt_util.

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

5. Execute o seguinte comando para fazer login no HSM. Substitua *<user name>* e *<password>* pelo nome do usuário e a senha do usuário de criptografia (CU).

```
Command: loginHSM -u CU -s <user name> -p <password>>
```

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 `genRSAKeyPair` 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 e um rótulo de *`tls_rsa_keypair`*.

  ```
  Command: genRSAKeyPair -m 2048 -e 65537 -l tls_rsa_keypair
  ```

  Se o comando tiver sido bem-sucedido, você verá a seguinte saída indicando que gerou com êxito um par de chaves RSA.

  ```
  Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

                  Cfm3GenerateKeyPair:    public key handle: 7    private key handle: 8

                  Cluster Status:
                  Node id 1 status: 0x00000000 : HSM Return: SUCCESS
  ```

- Para gerar uma chave privada EC em um HSM

  Use o comando `genECCKeyPair` para gerar um par de chaves EC. Este exemplo gera um par de chaves EC com um ID de curva de 2 (correspondente à curva `NID_X9_62_prime256v1`) e um rótulo de *`tls_ec_keypair`*.

  ```
  Command: genECCKeyPair -i 2 -l tls_ec_keypair
  ```

  Se o comando tiver sido bem-sucedido, você verá a seguinte saída indicando que você gerou com êxito um par de chaves EC.

  ```
  Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

                  Cfm3GenerateKeyPair:    public key handle: 7    private key handle: 8

                  Cluster Status:
                  Node id 1 status: 0x00000000 : HSM Return: SUCCESS
  ```

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.

> **Note**
>
> Arquivos de armazenamento de chaves gerados com o Key Management Utility (KMU) podem ser usadas com o Client SDK 3 e Client SDK 5.

Verifique o identificador da chave que corresponde à chave que você gostaria de exportar como um PEM falso e execute o comando a seguir para exportar a chave privada no formato PEM falso e salvá-la em um arquivo. Substitua os seguintes valores pelo seu próprio.

- *`<private_key_handle>`* Identificador da chave privada gerada. Esse identificador foi gerado por um dos comando de geração de chave na etapa anterior. No exemplo anterior, o identificador da chave privada é 8.

- *<web_server_fake_PEM.key>* Nome do arquivo no qual sua chave PEM falsa será gravada.

```
Command: getCaviumPrivKey -k <private_key_handle> -out <web_server_fake_PEM.key>
```

Exit

Execute o seguinte comando para interromper a key_mgmt_util.

```
Command: exit
```

Agora você deve ter um novo arquivo em seu sistema, localizado no caminho especificado por *<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 certificado SSL/TLS 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. Para instalar e configurar, faça um dos seguintes procedimentos:
   - the section called “Instalação do Mecanismo dinâmico do OpenSSL”
   - the section called “Mecanismo dinâmico do OpenSSL”

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). Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa. Substitua *<web_server.csr>* pelo nome do arquivo que contém a CSR.

   O comando `req` é interativo. Responda a cada campo. As informações do campo são copiadas para o certificado SSL/TLS.

   ```
   $ 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 o certificado do 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.

Importar uma chave privada e um certificado existentes

Pode ser que você já tenha uma chave privada e um certificado SSL/TLS correspondente que use para HTTPS em seu servidor Web. Se esse for o caso, você pode importar a chave para um HSM seguindo as etapas desta seção.

> **Note**
>
> Algumas observações sobre importações de chaves privadas e compatibilidade com o Client SDK:
>
> - A importação de uma chave privada existente requer o Client SDK 3.
> - Você pode usar chaves privadas do Client SDK 3 com o Client SDK 5.
> - O OpenSSL Dynamic Engine para Client SDK 3 não é compatível com as plataformas Linux mais recentes, mas a implementação do OpenSSL Dynamic Engine para Client SDK 5 sim. Você pode importar uma chave privada existente usando o Key Management Utility (KMU) fornecido com o Client SDK 3 e, em seguida, usar essa chave privada e a implementação do OpenSSL Dynamic Engine com o Client SDK 5 para suportar o descarregamento de SSL/TLS nas plataformas Linux mais recentes.

Para importar uma chave privada para um HSM com o Client SDK 3.

1. Conecte-se à sua instância do cliente Amazon EC2. Se necessário, copie a chave privada e o certificado existentes para a instância.
2. Instalar e configurar o Client SDK 3.
3. Execute 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
```

Ubuntu 20.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 22.04 LTS

```
$ sudo service cloudhsm-client start
```

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

4. Execute o comando a seguir para iniciar a ferramenta da linha de comando key_mgmt_util.

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

5. Execute o seguinte comando para fazer login no HSM. Substitua *<user name>* e *<password>* pelo nome do usuário e a senha do usuário de criptografia (CU).

```
Command: loginHSM -u CU -s <user name> -p <password>
```

6. Execute os comandos a seguir para importar sua chave privada em um HSM.

   a. Execute o comando a seguir para criar uma chave de encapsulamento simétrica que seja válida somente para a sessão atual. O comando e a saída são exibidos.

```
Command: genSymKey -t 31 -s 16 -sess -l wrapping_key_for_import

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
```

   b. Execute o comando a seguir para importar sua chave privada existente em um HSM. O comando e a saída são exibidos. Substitua os seguintes valores pelo seu próprio:

   - *<web_server_existing.key>* – Nome do arquivo que contém a chave privada.
   - *<web_server_imported_key>* – Rótulo da chave privada importada.
   - *<wrapping_key_handle>* – Identificador da chave de encapsulamento gerada no comando anterior. No exemplo anterior, o identificador de chave de encapsulamento é 6.

```
Command: importPrivateKey -f <web_server_existing.key> -
l <web_server_imported_key> -w <wrapping_key_handle>

BER encoded key length is 1219
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS
Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS
Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS
Private Key Unwrapped.  Key Handle: 8
Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

7. Execute o seguinte comando para exportar a chave privada no formato PEM falso e salvá-lo em um arquivo. Substitua os seguintes valores pelo seu próprio.

   - *<private_key_handle>* – Identificador da chave privada importada. Esse identificador foi gerado pelo segundo comando na etapa anterior. No exemplo anterior, o identificador da chave privada é 8.

- *<web_server_fake_PEM.key>* – Nome do arquivo que contém a chave privada PEM falsa exportada.

```
Command: getCaviumPrivKey -k <private_key_handle> -out <web_server_fake_PEM.key>
```

8. Execute o seguinte comando para interromper a key_mgmt_util.

```
Command: exit
```

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 de servidor web do Linux para descarregamento de SSL/TLS com o 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. Substitua *<web_server.crt>* pelo nome do seu certificado de 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. Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

```
$ sudo 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.

```
$ 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:

- Se estiver usando o Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se estiver usando o Client SDK 5

```
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:

- Se estiver usando o Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se estiver usando o Client SDK 5

```
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:

- Se estiver usando o Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se estiver usando o Client SDK 5

```
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 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:

- Se estiver usando o Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se estiver usando o Client SDK 5

```
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 cloudhsm;
    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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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
```

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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):

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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):

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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):

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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):

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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 do CU.

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 do CU.

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 do CU.

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 do CU.

Salve o arquivo.

Ubuntu 22.04 LTS

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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
```

CentOS 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
```

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

12. (Opcional) Configure sua plataforma para iniciar o NGINX na inicialização.

Amazon Linux

```
$ sudo chkconfig nginx on
```

Amazon Linux 2

```
$ 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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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
```

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Amazon Linux 2

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

CentOS 7

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

CentOS 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Red Hat 7

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Red Hat 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Ubuntu 16.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Ubuntu 18.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Ubuntu 20.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Substitua *<web_server.crt>* pelo nome do seu certificado de servidor Web.

Ubuntu 22.04 LTS

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

4. Copie sua chave privada PEM falsa para o local necessário para sua plataforma.

Amazon Linux

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Amazon Linux 2

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

CentOS 7

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

CentOS 8

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Red Hat 7

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Red Hat 8

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Ubuntu 16.04 LTS

```
$ sudo cp <web_server_fake_PEM.key> /etc/ssl/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Ubuntu 18.04 LTS

```
$ sudo cp <web_server_fake_PEM.key> /etc/ssl/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Ubuntu 20.04 LTS

```
$ sudo cp <web_server_fake_PEM.key> /etc/ssl/private/localhost.key
```

Substitua *<web_server_fake_PEM.key>* pelo nome do arquivo que contém a chave privada PEM falsa.

Ubuntu 22.04 LTS

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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.

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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.

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 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.

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 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.

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: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.

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 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 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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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
```

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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`.

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

Amazon Linux 2

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

CentOS 7

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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 do CU.

Red Hat 7

Use um editor de texto para editar o `/etc/sysconfig/httpd`.

- Se estiver usando o Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se estiver usando o Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Substitua *<CU user name>* e *<password>* pelas credenciais do CU.

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 do CU.

> **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 do CU.

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 do CU.

> **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 do CU.

> **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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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
```

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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

10. (Opcional) Configure sua plataforma para iniciar o Apache na inicialização.

Amazon Linux

```
$ sudo chkconfig httpd on
```

Amazon Linux 2

```
$ 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

O suporte para o OpenSSL Dynamic Engine ainda não está disponível.

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 descarga de SSL/TLS 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 descarga de SSL/TLS. 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 descarregamento SSL/TLS está usando o certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como OpenSSL s_client.

Para verificar o descarregamento de SSL/TLS 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 DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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. Garanta que o certificado SSL/TLS é aquele para o qual você configurou o servidor Web para usar.

Para verificar o descarregamento de SSL/TLS com OpenSSL s_client

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS. Substitua *<server name>* pelo nome DNS público ou endereço IP do seu servidor Web.

```
openssl s_client -connect <server name>:443
```

> **Tip**
>
> Você pode usar um serviço DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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 SSL/TLS é aquele para o qual você configurou o 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 (opcional).

Usando o Tomcat com JSSE para descarregamento de SSL/TLS no Linux

Este tópico fornece step-by-step instruções para configurar o descarregamento de SSL/TLS usando o Java Secure Socket Extension (JSSE) com o JCE SDK. AWS CloudHSM

Tópicos

Visão geral

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 de HSMs para esses servidores web. 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 para descarregamento de SSL/TLS no AWS CloudHSM Linux. Esses pré-requisitos devem ser atendidos para configurar o descarregamento SSL/TLS 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 Usar o Client SDK 5 para integração com Java Keytool e Jarsigner.

Depois de concluir essas etapas, vá para Etapa 2: Gerar ou importar uma chave privada e um certificado SSL/TLS.

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 ter mais informações, consulte Adicionar um HSM.
- 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 certificado SSL/TLS

Para habilitar o HTTPS, o servidor web Tomcat precisa de um certificado SSL/TLS e de uma chave privada correspondente. Para usar o servidor web SSL/TLS offload com 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. Em seguida, use a chave privada para criar uma solicitação de assinatura de certificado (CSR), que é então assinada para produzir um certificadoSSL/TLS.

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 descarregamento de SSL/TLS.

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>***: isto 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 atribuído.
- ***<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, SHA1withRSA, SHA224withRSA, SHA256withRSA, SHA384withRSA e SHA512withRSA).

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 certificado SSL/TLS 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. Essa 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 de servidor web do Linux para descarregamento de SSL/TLS com o 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.

Atualize o Classpath do Tomcat

1. Conecte-se à instância do cliente.
2. Localize a pasta de instalação do Tomcat.
3. Depois de substituir os dados *<VARIABLES>* abaixo pelos seus dados específicos, use o comando a seguir para adicionar a biblioteca Java e o caminho Java do Cloudhsm no caminho de classe do Tomcat, localizado no arquivo Tomcat/bin/catalina.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.

1. Vá até a pasta de instalação do Tomcat.
2. 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>***: isto 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 do Tomcat.

Iniciar o servidor

- Depois de substituir os dados *<VARIABLES>* abaixo pelos seus dados específicos, execute o seguinte comando 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 descarga de SSL/TLS 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 descarga de SSL/TLS. 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 descarregamento SSL/TLS está usando o certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como OpenSSL s_client.

Para verificar o descarregamento de SSL/TLS 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 DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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. Garanta que o certificado SSL/TLS é aquele para o qual você configurou o servidor Web para usar.

Para verificar o descarregamento de SSL/TLS com OpenSSL s_client

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS. Substitua *`<server name>`* pelo nome DNS público ou endereço IP do seu servidor Web.

   ```
   openssl s_client -connect <server name>:443
   ```

> **Tip**
>
> Você pode usar um serviço DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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 SSL/TLS é aquele para o qual você configurou o 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 (opcional).

Usar IIS com CNG para descarregamento de SSL/TLS no Windows

Este tutorial fornece step-by-step instruções para configurar o descarregamento de SSL/TLS AWS CloudHSM em um servidor web Windows.

Tópicos

Visão geral

No Windows, o aplicativo do servidor web Serviços de Informações da Internet (IIS) para Windows Server oferece suporte nativo a HTTPS. O AWS CloudHSM provedor de armazenamento de chaves (KSP) para a Cryptography API: Next Generation (CNG) da Microsoft fornece uma interface que permite que o IIS use os HSMs no cluster para descarregamento criptográfico 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

Para configurar o descarregamento SSL/TLS do servidor web 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 Conceitos básicos. 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 ter mais informações, consulte Adicionar um HSM.
3. Conecte-se ao servidor do Windows. Para obter mais informações, consulte Connect to Your Instance 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 Connect to Your Instance 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. Em seguida, 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. Em seguida, escolha Próximo.
9. Leia as informações sobre a função Servidor Web (IIS). Em seguida, escolha Próximo.
10. Em Select role services (Selecionar serviços de função), aceite os valores padrão ou altere as configurações como preferir. Em seguida, 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, o servidor web precisa de um certificado SSL/TLS e de uma chave privada correspondente. Para usar o descarregamento de SSL/TLS 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
- Obtenção e importação de um certificado assinado

Criar uma CSR

Use o AWS CloudHSM KSP em seu Windows Server para criar uma CSR.

Para criar uma CSR

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte Connect to Your Instance 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 o arquivo de solicitação de certificado, substitua *`IIS CertRequest .inf`* pelo nome de arquivo apropriado. Opcionalmente, você pode substituir *`IIS CertRequest .csr`* por um nome de arquivo diferente para seu arquivo CSR.

```
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 usar uma CA, envie o arquivo da CSR (`IISCertRequest.csr`) para ela e use a CA para criar um certificado SSL/TLS 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 substituir *`SelfSignedCA.key`* 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. Substitua *`SelfSignedCA.key`* pelo nome do arquivo que contém sua

chave privada (se for diferente). Opcionalmente, você pode substituir *SelfSignedCA.crt 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).
  - *IIS CertRequest .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 conterá o certificado assinado do 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 do servidor Web do Windows (IIS) para descarregamento de SSL/TLS com o 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 Connect to Your Instance 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 Certificates - Local Computer (Certificados – Computador Local), clique duas vezes em Trusted Root Certification Authorities (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 Connect to Your Instance 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. Substitua *IISCert.crt* pelo nome do arquivo que contém o certificado assinado do 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 descarga de SSL/TLS 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 descarga de SSL/TLS. 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 descarregamento SSL/TLS está usando o certificado autoassinado. Faça isso com um navegador da Web ou com uma ferramenta como OpenSSL s_client.

Para verificar o descarregamento de SSL/TLS 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 DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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. Garanta que o certificado SSL/TLS é aquele para o qual você configurou o servidor Web para usar.

Para verificar o descarregamento de SSL/TLS com OpenSSL s_client

1. Execute o seguinte comando OpenSSL para se conectar ao seu servidor Web usando HTTPS. Substitua *<server name>* pelo nome DNS público ou endereço IP do seu servidor Web.

```
openssl s_client -connect <server name>:443
```

> **Tip**
>
> Você pode usar um serviço DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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 SSL/TLS é aquele para o qual você configurou o 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 (opcional).

Adicione um balanceador de carga com o Elastic Load Balancing (opcional)

Depois de configurar o descarregamento de SSL/TLS com um servidor Web, você pode criar mais servidores Web e um balanceador de carga no Elastic Load Balancing que roteia o tráfego HTTPS aos 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

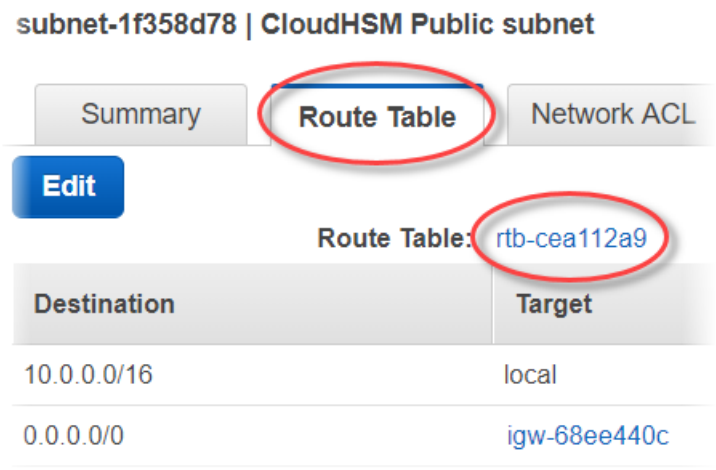
- Criar uma sub-rede para um segundo servidor Web
- Criar o segundo servidor Web
- Criar o balanceador de carga

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 bloco CIDR IPv4, 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.



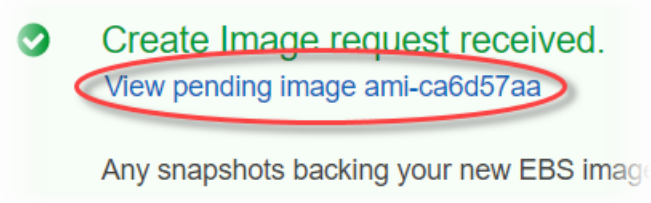
6. Marque a caixa de seleção ao lado da tabela de rotas.
7. Escolha a guia Associações de sub-redes. Em seguida, escolha Editar.
8. Marque a caixa de seleção ao lado da sub-rede pública que você criou anteriormente neste procedimento. Em seguida, escolha Salvar.

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.

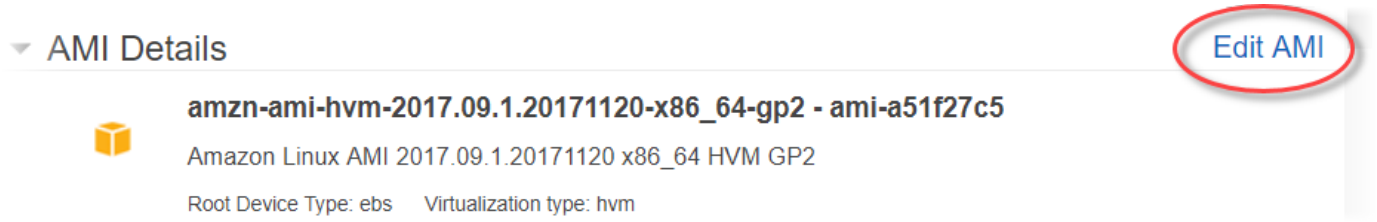
Para criar um segundo servidor Web

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 link View pending image ami-***<AMI ID>***.



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.



9. No painel de navegação esquerdo, escolha Meus AMIs. Em seguida, limpe o texto na caixa de pesquisa.
10. Ao lado de sua imagem do servidor Web, escolha Select.
11. Escolha Sim, quero continuar neste 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. Selecione esse grupo de segurança para permitir que a instância do servidor Web se conecte a 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 possuo acesso ao arquivo de chave privada selecionado (***<private key file name>***.pem) e que, sem esse arquivo, não será possível fazer o login na 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.

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 load balancer

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. Selecione Next: Review (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 poderá verificar se ele está enviando tráfego HTTPS aos seus servidores Web com descarregamento de SSL/TLS com o 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 DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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. Substitua *<DNS name>* pelo nome DNS do seu balanceador de carga.

```
openssl s_client -connect <DNS name>:443
```

> **Tip**
>
> Você pode usar um serviço DNS, como o Amazon Route 53 para rotear o nome de domínio do site (por exemplo, https://www.example.com/) para seu servidor da 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.

Configurar o Windows Server como uma autoridade de certificação (CA) com o AWS CloudHSM

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 seu servidor Windows ao seu AWS CloudHSM cluster. Na última etapa, você assina uma solicitação de assinatura de certificado (CSR) com a CA do Windows Server.

Para obter mais informações, consulte os tópicos a seguir.

Tópicos

Etapa 1 da CA do Windows Server: 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 Conceitos básicos. 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 ter mais informações, consulte Adicionar um HSM.
3. Conecte-se à instância do cliente. Para obter mais informações, consulte Connect to Your Instance no Guia do usuário do Amazon EC2.
4. Crie um usuário de criptografia (CU) usando Gerenciamento de usuários do HSM com a CLI do CloudHSM 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.
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 criar uma CA do Windows Server com AWS CloudHSM, acesseCriar CA do Windows Server.

Etapa 2 do CA do Windows Server: Criar um CA do Windows Server com o 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

1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte Connect to Your Instance 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. Em seguida, 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. Em seguida, escolha Próximo.
13. Em Role Services, selecione Certification Authority. Em seguida, escolha Próximo.
14. Em Setup Type, selecione Standalone CA. Em seguida, escolha Próximo.
15. Em CA Type, selecione Root CA. Em seguida, 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. Em seguida, 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. Em seguida, 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. Em seguida, 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 da CA do Windows Server: 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

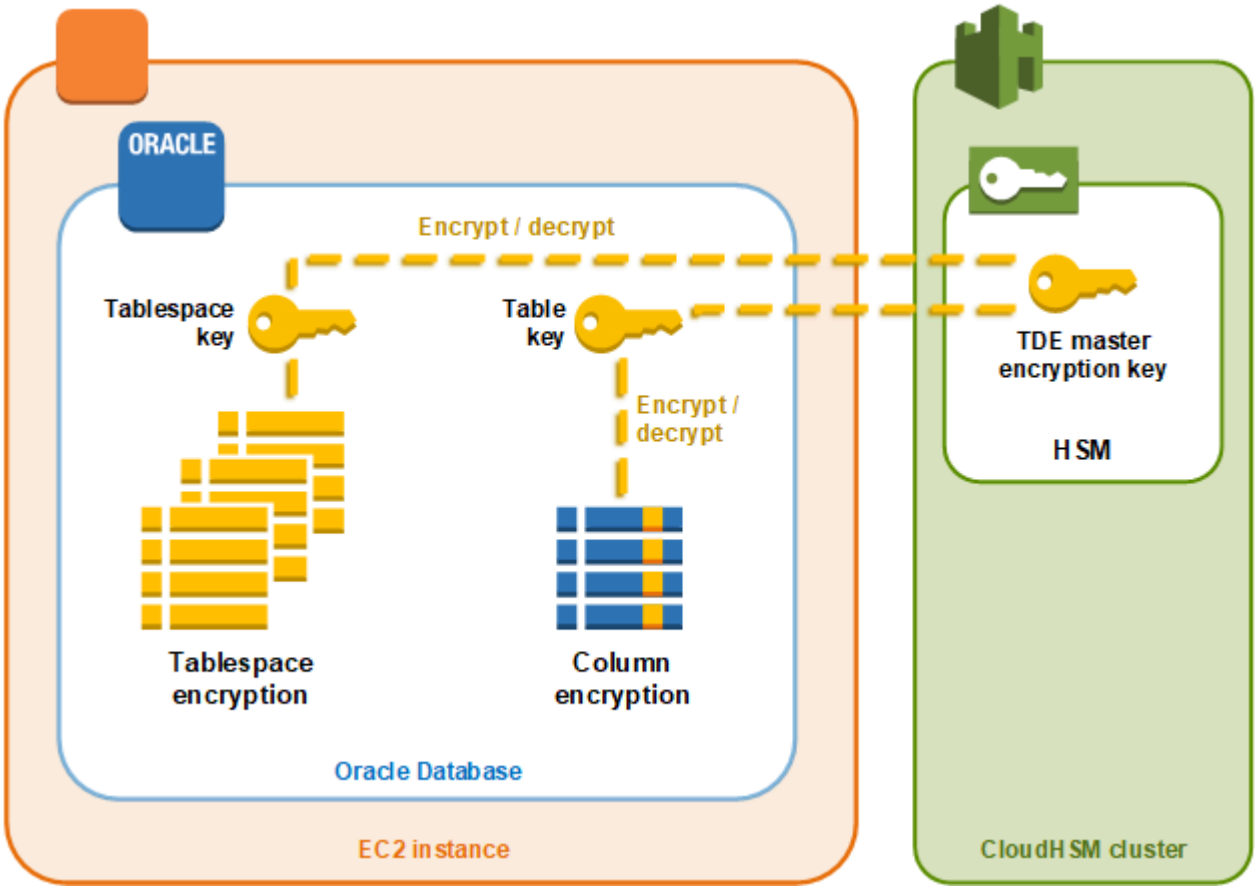
1. Se você ainda não tiver feito isso, conecte-se ao Windows Server. Para obter mais informações, consulte Connect to Your Instance no Guia do usuário do Amazon EC2.
2. No Windows Server, inicie o Gerenciador de servidores.
3. No painel Server Manager, no canto superior direito, escolha Tools, Certification 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.

Oracle Database Transparent Data Encryption (TDE) com o 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 nos HSMs do 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 software AWS CloudHSM para PKCS #11 para armazenar a chave-mestra do TDE nos HSMs do 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.

Para configurar a integração do Oracle TDE com AWS CloudHSM

1. Siga as etapas em Configurar pré-requisitos para preparar o ambiente.
2. Siga as etapas Configurar o banco de dados para configurar o Oracle Database para se integrar ao seu AWS CloudHSM cluster.

Oracle TDE com AWS CloudHSM: 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 de criptografia (CU) para possuir e gerenciar a chave de criptografia mestre do TDE nos 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 Conceitos básicos. 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 ter mais informações, consulte Adicionar um HSM.
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 Gerenciamento de usuários de HSM.

Depois de concluir essas etapas, você poderá Configurar o banco de dados.

Oracle TDE com AWS CloudHSM: Configure o banco de dados e gere a chave mestra de criptografia

Para integrar o Oracle TDE ao seu AWS CloudHSM cluster, consulte os tópicos a seguir:

1. Atualizar a configuração do Oracle Database para usar os HSMs no seu cluster como o módulo de segurança externo. Para obter informações sobre módulos de segurança externos, consulte o tópico de Introdução ao Transparent Data Encryption no Oracle Database Advanced Security Guide.
2. Gerar a chave de criptografia mestra do Oracle TDE nos HSMs no seu cluster.

Atualizar a configuração do Oracle Database

Para atualizar a configuração do Oracle Database para usar um HSM no cluster como o módulo de segurança externa, conclua as etapas a seguir. Para obter informações sobre módulos de segurança externos, consulte o tópico de Introdução ao Transparent Data Encryption no Oracle Database Advanced Security Guide.

Para atualizar a configuração Oracle

1. Conecte-se à sua instância do cliente Amazon EC2. Esta é a instância na qual você instalou o Oracle Database.
2. Faça uma cópia de backup do arquivo chamado `sqlnet.ora`. Para a localização deste arquivo, consulte a documentação da Oracle.
3. Use um editor de texto para editar o arquivo chamado `sqlnet.ora`. Adicione a seguinte linha. Se uma linha existente no arquivo começar com `encryption_wallet_location`, substitua-a pela seguinte.

```
encryption_wallet_location=(source=(method=hsm))
```

Salve o arquivo.

4. Execute o comando a seguir para criar o diretório em que o Oracle Database espera encontrar o arquivo de biblioteca da biblioteca de software AWS CloudHSM PKCS #11.

```
sudo mkdir -p /opt/oracle/extapi/64/hsm
```

5. Execute o comando a seguir para copiar a biblioteca de AWS CloudHSM software do arquivo PKCS #11 para o diretório que você criou na etapa anterior.

```
sudo cp /opt/cloudhsm/lib/libcloudhsm_pkcs11.so /opt/oracle/extapi/64/hsm/
```

> **Note**
>
> O diretório `/opt/oracle/extapi/64/hsm` deve conter apenas um arquivo de biblioteca. Remova qualquer outro arquivo existente nesse diretório.

6. Execute o seguinte comando para alterar a propriedade do diretório `/opt/oracle` e tudo dentro dele.

```
sudo chown -R oracle:dba /opt/oracle
```

7. Inicie o Oracle Database.

Gerar a chave de criptografia mestra do Oracle TDE

Para gerar a chave-mestra do Oracle TDE nos HSMs do seu cluster, conclua as etapas no 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 PCKS #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. Substitua *<CU user name>* pelo nome do usuário de criptografia (CU). Substitua *<password>* pela senha CU.

> **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. Substitua *<CU user name>* pelo nome do usuário de criptografia (CU). Substitua *<password>* pela senha CU.

- 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

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

Microsoft SignTool com AWS CloudHSM a 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.

Microsoft SignTool com a AWS CloudHSM 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 Connect to Your Instance 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 outros aplicativos.

   ```
   certreq.exe -accept signedCertificate.cer
   ```

Agora, você pode usar o certificado importado para Assinar um arquivo.

Microsoft SignTool com AWS CloudHSM a etapa 3: assinar um arquivo

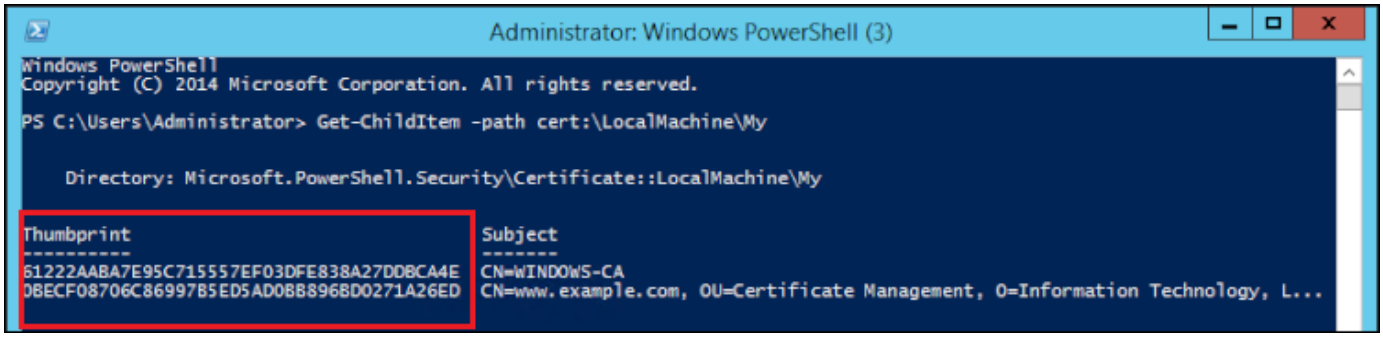
Agora você está pronto para usar SignTool seu certificado importado para assinar seu arquivo de exemplo. Para fazer isso, você precisa saber o hash SHA-1 do certificado ou a impressão digital. A impressão digital é usada para garantir que use SignTool apenas certificados verificados por. AWS CloudHSM 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:

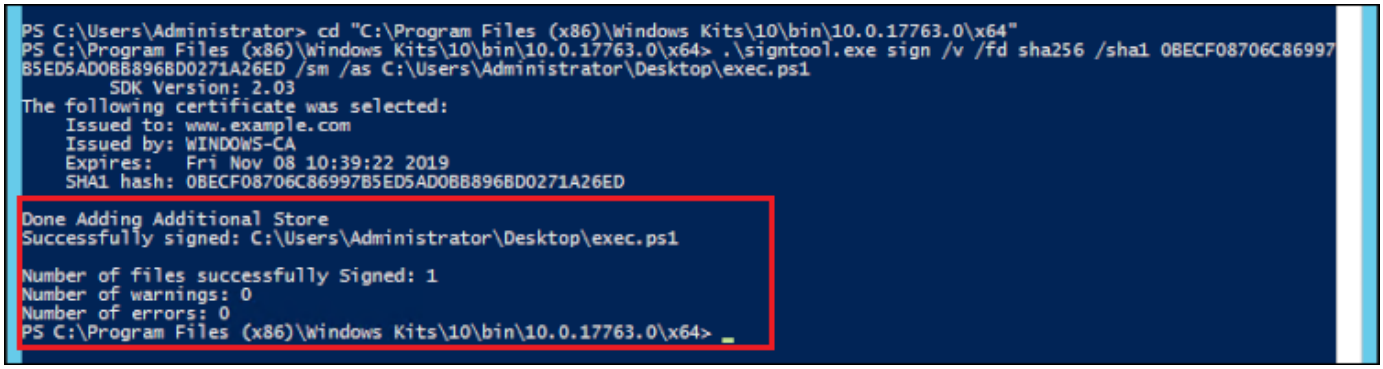
   ```
   Get-ChildItem -path cert:\LocalMachine\My
   ```

   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
   ```



4. (Opcional) Para verificar a assinatura no arquivo, use o seguinte comando:

```
signtool.exe verify /v /pa C:\Users\Administrator\Desktop\<test>.ps1
```

Java Keytool e Jarsigner

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:

- Usar o Client SDK 5 para integração com Java Keytool e Jarsigner
- Usar o Client SDK 3 para integração com Java Keytool e Jarsigner

Usar o Client SDK 5 para integração com Java Keytool e Jarsigner

AWS CloudHSM O armazenamento de chaves é um armazenamento de chaves JCE para fins especiais que utiliza certificados associados às chaves em seu 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 usar o armazenamento de AWS CloudHSM chaves, primeiro você deve inicializar e configurar o SDK do AWS CloudHSM JCE.

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.

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 o keytool com o armazenamento de chave do AWS CloudHSM pode causar erros.

Usando o armazenamento de AWS CloudHSM chaves com o keytool

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 CLOUDHSM -J-classpath< '-J/opt/cloudhsm/java/*'>
```

Windows

```
-storetype CLOUDHSM -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, consulteUsando AWS CloudHSM KeyStore. 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.

Criar novas chaves com keytool

Você pode usar keytool para gerar os tipos de chaves RSA, AES e DESede suportadas pelo SDK JCE do AWS CloudHSM.

> **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 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 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 “my_keystore.store”, no diretório de trabalho. Substitua *<secret label>* por um rótulo exclusivo.

Linux

```
$ keytool -genseckey -alias <secret label> -keyalg aes \
 -keysize 256 -keystore my_keystore.store \
 -storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
```

Windows

```
PS C:\> keytool -genseckey -alias <secret label> -keyalg aes `
 -keysize 256 -keystore my_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 “my_keystore.store”, no diretório de trabalho. Substitua *<RSA key pair label>* por um rótulo exclusivo.

Linux

```
$ keytool -genkeypair -alias <RSA key pair label> \
 -keyalg rsa -keysize 2048 \
 -sigalg sha512withrsa \
 -keystore my_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 my_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.

Excluir uma chave usando 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();
```

Gerar um CSR usando keytool

Você recebe a maior flexibilidade na geração de um pedido de assinatura de certificado (CSR) se você usar o Mecanismo dinâmico do OpenSSL. O comando a seguir usa o keytool para gerar um CSR para um par de chaves com o alias `my-key-pair`.

Linux

```
$ keytool -certreq -alias <key pair label> \
 -file my_csr.csr \
 -keystore my_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -certreq -alias <key pair label> `
 -file my_csr.csr `
 -keystore my_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 “Usando o keytool para importar certificados intermediários e raiz para o armazenamento de AWS CloudHSM chaves ”

Usando o keytool para importar certificados intermediários e raiz para o armazenamento de AWS CloudHSM chaves

Para importar um certificado CA, 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 my_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -import -trustcacerts -alias rootCAcert `
 -file rootCAcert.cert -keystore my_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.

Usando o keytool para excluir certificados do armazenamento de AWS CloudHSM chaves

O comando a seguir mostra um exemplo de como excluir um certificado de um repositório de chaves Java keytool.

Linux

```
$ keytool -delete -alias mydomain \
 -keystore my_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -delete -alias mydomain `
 -keystore my_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.

Importando 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 my_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 my_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 usando Keytool

O exemplo a seguir gera um certificado no formato binário X.509. Para exportar um certificado legível por humanos, adicione `-rfc` ao comando `-exportcert`.

Linux

```
$ keytool -exportcert -alias <key pair label> \
 -file my_exported_certificate.crt \
 -keystore my_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 my_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Usando o armazenamento de AWS CloudHSM chaves com o Jarsigner

O Jarsigner é um utilitário de linha de comando popular para assinar arquivos JAR usando uma chave armazenada com segurança em um HSM. Um tutorial completo sobre Jarsigner está fora do escopo da documentação do AWS CloudHSM . 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.

Configuração de chaves e certificados

Antes de assinar arquivos JAR com Jarsigner, verifique se você configurou ou concluiu 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 com keytool”. Se você usar várias instâncias de cliente para assinar seus 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 “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 the section called “Usando o keytool para importar certificados intermediários e raiz para o armazenamento de AWS CloudHSM chaves ”.

Assine um arquivo JAR usando AWS CloudHSM e Jarsigner

Use o seguinte comando para assinar um arquivo JAR:

Linux;

Para OpenJDK 8

```
jarsigner -keystore my_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 my_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 OpenJDK8

```
jarsigner -keystore my_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 my_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 OpenJDK8

```
jarsigner -verify \
 -keystore my_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 my_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 my_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 my_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

1. Não oferecemos suporte a chaves EC com Keytool e Jarsigner.

Usar o Client SDK 3 para integração com Java Keytool e Jarsigner

AWS CloudHSM O armazenamento de chaves é um armazenamento de chaves JCE para fins especiais que utiliza certificados associados às chaves em seu 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 usar o armazenamento de AWS CloudHSM chaves, primeiro você deve inicializar e configurar o SDK do AWS CloudHSM JCE.

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.

Usando o armazenamento de AWS CloudHSM chaves com o keytool

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, consulteUsando AWS CloudHSM KeyStore. 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.

Criar novas chaves com keytool

Você pode usar o keytool para gerar qualquer tipo de chave compatível com o JCE AWS CloudHSM SDK. 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 HSM e usá-las com o keytool ou o Jarsigner são mostradas na amostra de código em Registrando chaves pré-existentes com o 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 “my_keystore.store”, no diretório de trabalho. Substitua *<secret label>* por um rótulo exclusivo.

```
keytool -genseckey -alias <secret label> -keyalg aes \
  -keysize 256 -keystore my_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 “my_keystore.store”, no diretório de trabalho. Substitua *<RSA key pair label>* por um rótulo exclusivo.

```
keytool -genkeypair -alias <RSA key pair label> \
        -keyalg rsa -keysize 2048 \
        -sigalg sha512withrsa \
        -keystore my_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 “my_keystore.store”, no diretório de trabalho. Substitua *<ec key pair label>* por um rótulo exclusivo.

```
keytool -genkeypair -alias <ec key pair label> \
        -keyalg ec -keysize 256 \
        -sigalg SHA512withECDSA \
        -keystore my_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.

Excluir uma chave usando 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,deleteKey.

Gerar um CSR usando keytool

Você recebe a maior flexibilidade na geração de um pedido de assinatura de certificado (CSR) se você usar o Mecanismo dinâmico do OpenSSL. O comando a seguir usa o keytool para gerar um CSR para um par de chaves com o alias `my-key-pair`.

```
keytool -certreq -alias <key pair label> \
        -file my_csr.csr \
        -keystore my_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.

Usando o keytool para importar certificados intermediários e raiz para o armazenamento de AWS CloudHSM chaves

Para importar um certificado CA, 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 my_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.

Usando o keytool para excluir certificados do armazenamento de AWS CloudHSM chaves

O comando a seguir mostra um exemplo de como excluir um certificado de um repositório de chaves Java keytool.

```
keytool -delete -alias mydomain -keystore \
        -keystore my_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.

Importando 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 my_certificate.crt \
        -keystore my_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, você deve primeiro importar os metadados de chave e certificado para o repositório de chaves. Para obter instruções sobre como importar os metadados do certificado, consulte o exemplo de código em Registrando chaves pré-existentes com o Key Store. AWS CloudHSM

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 usando Keytool

O exemplo a seguir gera um certificado no formato binário X.509. Para exportar um certificado legível por humanos, adicione `-rfc` ao comando `-exportcert`.

```
keytool -exportcert -alias <key pair label> \
        -file my_exported_certificate.crt \
        -keystore my_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Usando o armazenamento de AWS CloudHSM chaves com jarsigner

O Jarsigner é um utilitário de linha de comando popular para assinar arquivos JAR usando uma chave armazenada com segurança em um HSM. Um tutorial completo sobre Jarsigner está fora do

escopo da documentação do AWS CloudHSM . 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.

Configuração de chaves e certificados

Antes de assinar arquivos JAR com Jarsigner, verifique se você configurou ou concluiu as seguintes etapas:

1. Siga as orientações nos pré-requisitos de repositório de chaves AWS CloudHSM.
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. Use o exemplo 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 Criar novas chaves com keytool. Se você usar várias instâncias de cliente para assinar seus 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 seguinte comando para assinar um arquivo JAR:

```
jarsigner -keystore my_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 my_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

A lista a seguir fornece a lista atual de problemas conhecidos.

- 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 de keytool e Jarsigner, o provedor SunEC precisa ser removido/desativado da 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.

Registrando chaves pré-existentes com AWS CloudHSM o armazenamento de chaves

Para maior segurança e flexibilidade em atributos e rotulagem, recomendamos a geração de chaves de 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.Enumeration;

//
// 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\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 alises 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[] nameOid, String value) {
         byte[] name = null;
         name = Encoder.encodeSet(
                     Encoder.encodeSequence(
                             Encoder.encodeOid(nameOid),
                             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());
        }
    }

}
```

Outras integrações de fornecedores terceiros

Vários fornecedores terceirizados oferecem suporte AWS CloudHSM como base 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 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 suporta 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

Trabalhar com logs do Client SDK

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

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:

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
```

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 logs depende do sistema operacional da instância do cliente do Amazon EC2 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.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. Você pode visualizar, pesquisar e baixar eventos recentes em sua conta da AWS . 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 mais informações, 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` `eListTags`, 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:

- 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 o elemento CloudTrail UserIdentity.

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": "AROAJZVM5NEGZSTCITAMM:ExampleSession",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/AdminRole/ExampleSession",
        "accountId": "111122223333",
        "accessKeyId": "ASIAIY22AX6VRYNBGJSA",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2017-07-11T03:48:44Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAJZVM5NEGZSTCITAMM",
                "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-6lz2hfmnzbx",
            "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 nos CloudWatch SDKs AWS CLIe Logs.

Tópicos

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. Os logs de auditoria de vários HSMs, mesmo aqueles no mesmo cluster, podem ser diferentes. Por exemplo, apenas o primeiro HSM em cada cluster registra a inicialização do HSM. Os eventos de inicialização não são exibidos nos logs de HSMs clonados de 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 eles recebem a chave por meio da 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.

Visualizando registros de auditoria do HSM 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.

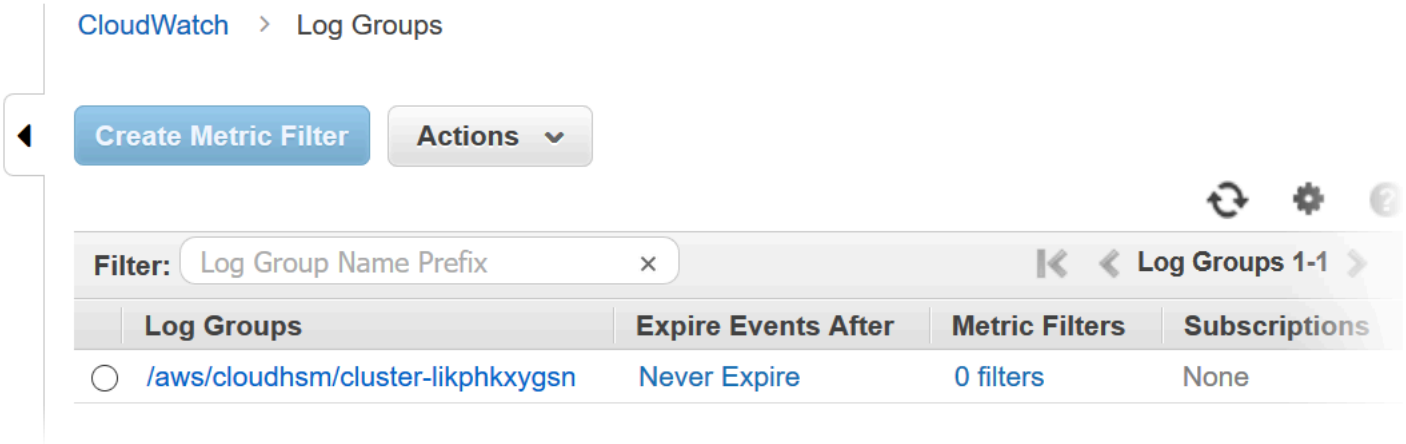
- O nome do grupo de logs é `/aws/cloudhsm/<cluster ID>`; por exemplo, `/aws/cloudhsm/cluster-likphkxygsn`. Ao usar o nome do grupo de registros em um PowerShell comando AWS CLI or, certifique-se de colocá-lo entre aspas duplas.
- O nome do fluxo de logs é o ID do HSM; por exemplo, `hsm-nwbbiqbj4jk`.

  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.

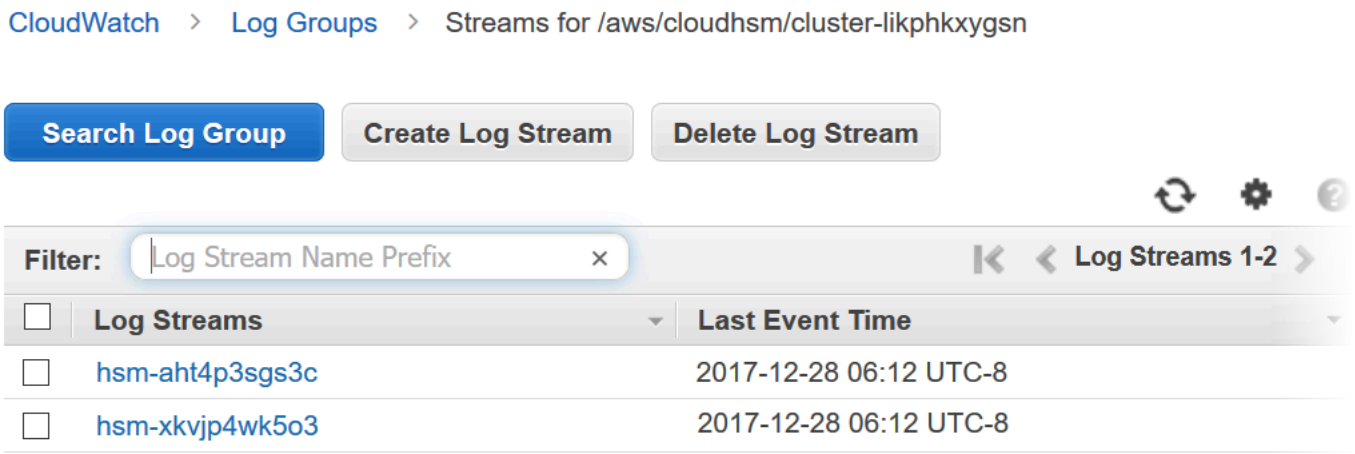
Para obter mais informações sobre CloudWatch os conceitos de registros, consulte Conceitos no Guia do usuário do Amazon CloudWatch Logs.

Você pode ver os registros de auditoria de um HSM na página CloudWatch Logs, nos comandos CloudWatch Logs no AWS Management Console, nos PowerShell cmdlets CloudWatch Logs ou nos SDKs de Logs. AWS CLI CloudWatch Para obter instruções, consulte Exibir dados de registro no Guia do usuário do Amazon CloudWatch Logs.

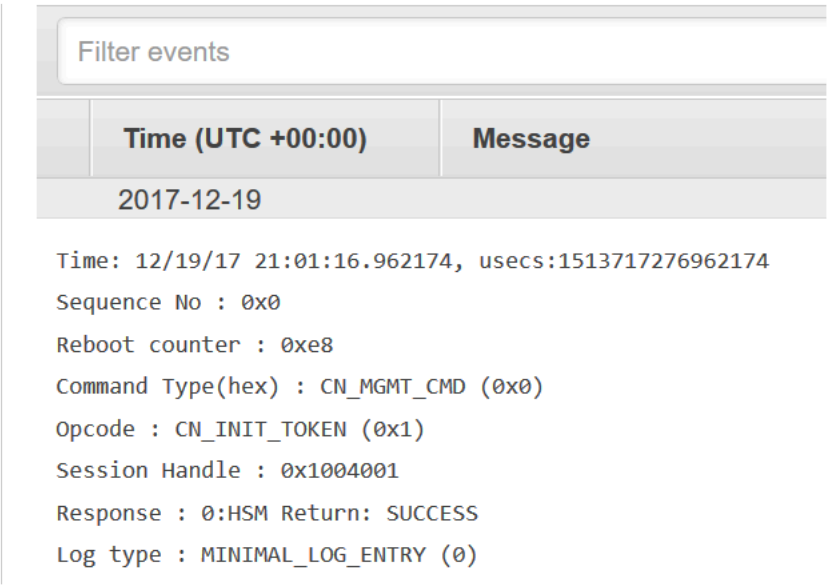
Por exemplo, a imagem a seguir mostra o grupo de logs do cluster do `cluster-likphkxygsn` no AWS Management Console.



Ao escolher o nome do grupo de logs do cluster, você pode visualizar o fluxo de logs de cada um dos HSMs no cluster. A imagem a seguir mostra os fluxos de logs dos HSMs no cluster do `cluster-likphkxygsn`.

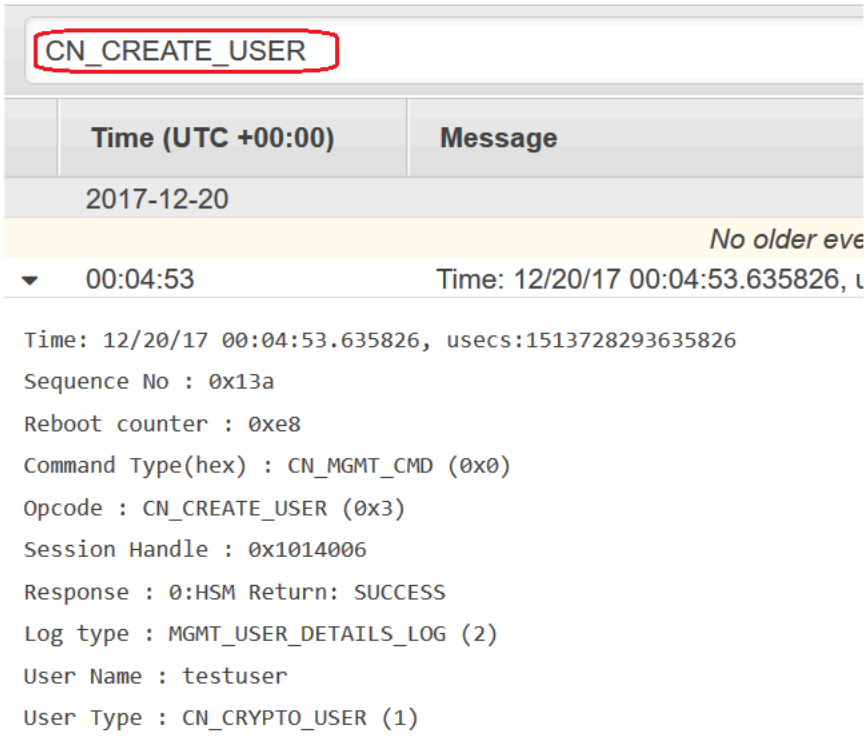


Ao escolher um nome do fluxo de logs do HSM, você pode visualizar os eventos no log de auditoria. Por exemplo, este evento, que tem um número de sequência de 0x0 e um `Opcode` de `CN_INIT_TOKEN`, geralmente é o primeiro evento do primeiro HSM em cada cluster. Ele registra a inicialização do HSM no cluster.



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.



Interpretar logs de auditoria do HSM

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.

Tempo

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, consulteReferência do log de auditoria HSM.

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 log para analisar o ciclo de vida dos HSMs e obter informações sobre a 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

O fluxo de logs de auditoria para o primeiro HSM em cada cluster difere significativamente dos fluxos de log de outros HSMs no cluster. O log de auditoria do primeiro HSM em cada cluster registra a criação e a inicialização. Os logs de outros 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 HSMs subsequentes no mesmo cluster são 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_CRYPTO_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, os outros HSMs no cluster também registram um evento de login para o usuário. Você pode encontrar os eventos de login correspondentes nos fluxos de logs 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_CRYPTO_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_CRYPTO_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_CRYPTO_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_CRYPTO_USER (1)
```

Exemplo: criar e excluir usuários

Este exemplo mostra os eventos de log registrados quando um responável 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_CRYPTO_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_CRYPTO_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_CRYPTO_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_CRYPTO_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_CRYPTO_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_CRYPTO_USER (1)
```

Exemplo: gerar e sincronizar uma chave

Este exemplo mostra o efeito da criação de uma chave em um cluster com vários HSMs. A chave é gerada em um HSM, extraída do HSM como um objeto mascarado e inserida nos outros HSMs como um objeto mascarado.

> **Note**
>
> As ferramentas de cliente podem falhar ao sincronizar a chave. Ou o comando pode incluir o parâmetro min_srv, que sincroniza a chave apenas para o número especificado de HSMs. Em ambos os casos, o AWS CloudHSM serviço sincroniza a chave com os outros HSMs no cluster. Como os HSMs registram apenas comandos de gerenciamento do lado do cliente em seus logs, a sincronização do lado do servidor não é registrada no log 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_CRYPTO_USER (1)
```

A UC executa um exSymKeycomando 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_CRYPTO_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 em ambos os HSMs 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 de log de auditoria que são registrados quando um usuário de criptografia exporta chaves de um cluster com vários 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_CRYPTO_USER (1)
```

A UC executa um exSymKeycomando para exportar a chave7, uma chave AES de 256 bits. O comando usa a chave 6, uma chave AES de 256 bits nos HSMs, como a chave de encapsulamento.

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 logs 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 nos HSMs em um cluster. Neste exemplo, o usuário criptográfico (CU) usa o imSymKeycomando para importar uma chave AES para os 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 de cliente tentam automaticamente sincronizar a nova chave com outros 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 logs 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)
```

Referência do log de auditoria HSM

AWS CloudHSM registra os comandos de gerenciamento do HSM em eventos de registro de auditoria. Cada evento tem um valor do código de operação (`0pcode`) que identifica a ação ocorrida e sua resposta. Você pode usar os valores `0pcode` para pesquisar, classificar e filtrar os logs.

A tabela a seguir define os `0pcode` valores em um registro AWS CloudHSM de auditoria.

Código de operação (Opcode)	Descrição
Login do usuário: esses eventos incluem o nome do usuário e o tipo de usuário	
`CN_LOGIN (0xd)`	Login de usuário
`CN_LOGOUT (0xe)`	Sair do usuário
`CN_APP_FINALIZE`	A conexão com o HSM foi encerrada. Todas as chaves de sessão ou tokens de quorum 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
`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

Código de operação (Opcode)	Descrição
`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 (unwrapKeyy)
`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.
Back up HSMs	
`CN_BACKUP_BEGIN`	Comece o processo de backup
`CN_BACKUP_END`	Concluiu o processo de backup
`CN_RESTORE_BEGIN`	Comece a restaurar a partir de um backup

Código de operação (Opcode)	Descrição
`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	
`CN_FIPS_RAND`	Gere um número aleatório compatível com FIPS

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	Descrição
HsmUnhealthy	A instância do HSM não está funcionando corretamente. AWS CloudHSM substitui automaticamente as instâncias não íntegras para você. Você pode

Métrica	Descrição
	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.
HsmSslCtxsOccupied [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. AWS CloudHSM também pode ter até duas conexões abertas com o HSM para monitorar a integridade dos 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. Atualmente, isso é 1.024.
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 hsm2m.medium.

AWS CloudHSM Desempenho

Para clusters de produção, você deve ter pelo menos duas instâncias do HSM distribuídas em duas 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

- [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 workload exceder a capacidade do HSM do cluster, você receberá mensagens de erro informando que os HSMs estão ocupados ou limitados. 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 uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. 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 Serviços da AWS no escopo do programa de conformidade .
- 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

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 seu conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas Frequentes sobre Privacidade de Dados.. Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and GDPR 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 os recursos. AWS Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail.
- 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 sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para ter mais informações sobre endpoints do FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-2.

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações 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 os SDKs. AWS CLI Quaisquer dados inseridos em tags ou 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, recomendemos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse 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 ter mais informações, consulte AWS CloudHSM backups de cluster.

Criptografia em trânsito

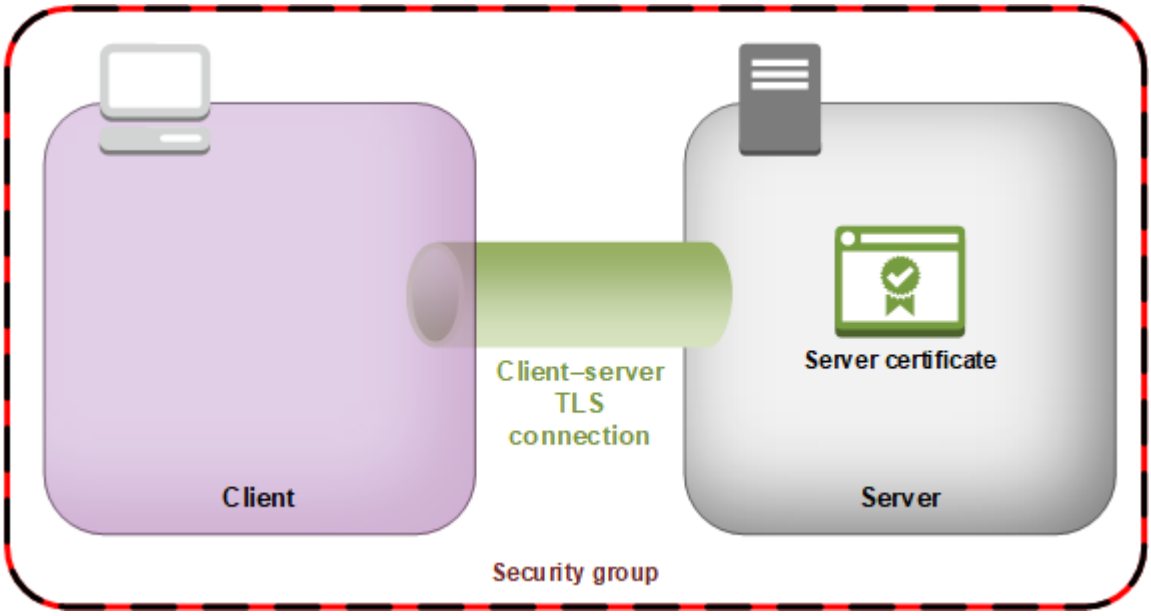
A comunicação entre o AWS CloudHSM cliente e o HSM em seu cluster é criptografada de ponta a ponta. Esta comunicação só pode ser descriptografada pelo seu cliente e seus HSMs. Para ter mais informações, consulte nd-to-end Criptografia E.

AWS CloudHSM end-to-end criptografia de cliente

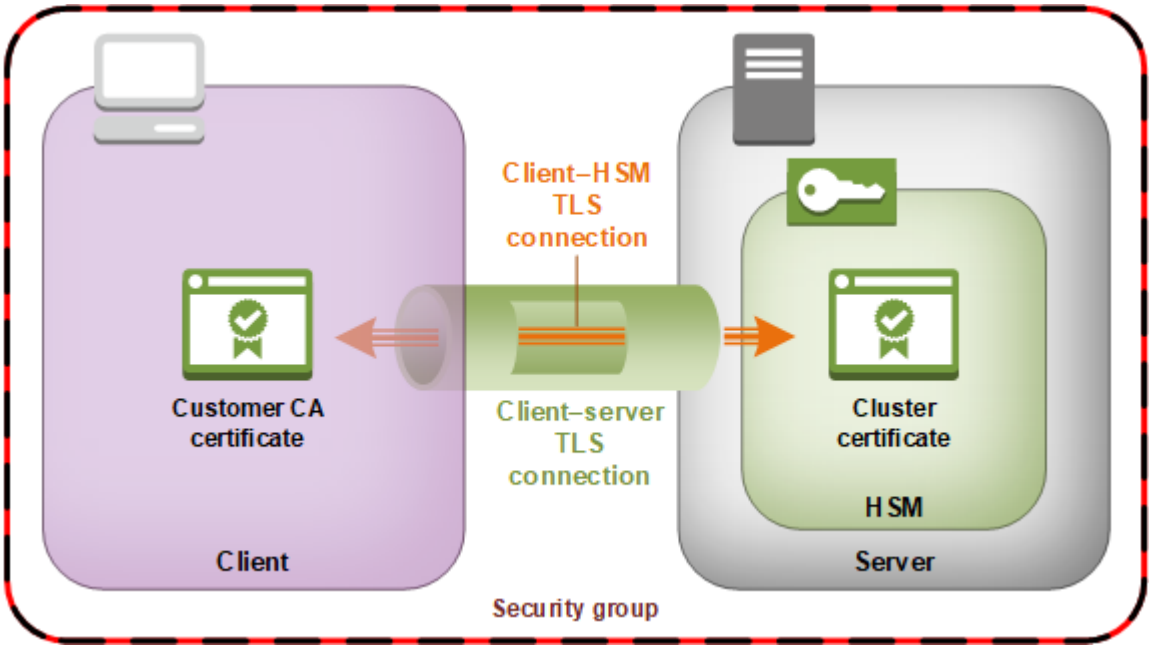
A comunicação entre a instância do cliente e os HSMs no cluster é criptografada de ponta a ponta. Somente o cliente e seus HSMs podem descriptografar a comunicação.

O processo a seguir explica como o cliente estabelece comunicação end-to-end criptografada com um HSM.

1. O cliente estabelece uma conexão de Segurança da camada de transporte (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 HSMcliente 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

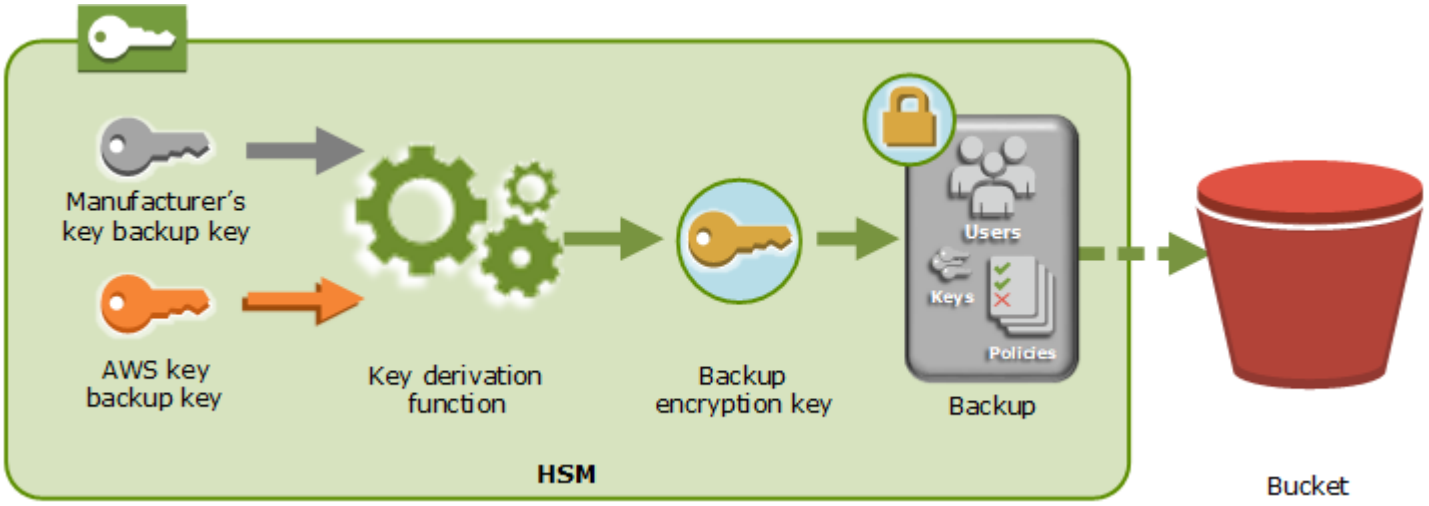
Ao fazer AWS CloudHSM um backup do HSM, o HSM criptografa todos os seus dados antes de enviá-los para. AWS CloudHSM Os dados nunca saem do HSM em formato de texto simples. Além disso, os backups não podem ser descriptografados AWS porque AWS não tem acesso à chave usada para descriptografar os backups.

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 somente em HSMs AWS de propriedade própria, 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.

Quando você anexa 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;

Quando você anexa 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. Cada instrução é estruturada como mostrado no exemplo a seguir.

```
{
  "Version": "2012-10-17",
  "Statement":[{
    "Effect": "effect",
    "Action": "action",
    "Resource": "resource-arn",
    "Condition": {
      "condition": {
        "key":"value"
      }
    }
  }]
}
```

- 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 ter mais informações, consulte Chaves de condição para AWS CloudHSM.

Para obter mais informações, consulte o Guia 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 :

- AWSCloudHSMFullAccess— Concede 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, é possível criar um grupo de usuários que você usa para conceder acesso somente leitura a 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 recurso 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

Example 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. Ela não permite que o usuário exclua clusters ou HSMs.

```
{
   "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. Ela 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 AWSServiceRoleForCloudHSMvinculada ao serviço em sua conta. Essa função permite AWS CloudHSM registrar eventos. Para ter 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.

```
{
   "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 AWSServiceRoleForCloudHSMvinculada ao serviço em sua conta.

Essa função permite AWS CloudHSM registrar eventos. Para ter mais informações, consulte Funções vinculadas a serviços para AWS CloudHSM.

```
{
   "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. AWSServiceRoleForCloudHSM 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.

```
{
    "Version": "2018-06-12",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        }
    ]
}
```

A política de confiança da AWSServiceRoleForCloudHSMfunção permite que você assuma AWS CloudHSM a função.

```
{
  "Version": "2018-06-12",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudhsm.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Criar uma função vinculada a serviços (automática)

AWS CloudHSM cria a AWSServiceRoleForCloudHSMfunção 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 AWSServiceRoleForCloudHSMfunção, você pode usar o console, o comando create-cluster ou a operação da CreateClusterAPI para criar um cluster. Em seguida, use o console, o comando delete-cluster ou a operação da DeleteClusterAPI 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. Para obter mais informações, consulte a seção a seguir.

> **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 AWSServiceRoleForCloudHSMfunção.

Criar uma função vinculada a serviços (manual)

Você pode usar o console do IAM ou a API para criar a AWSServiceRoleForCloudHSMfunção. 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 AWSServiceRoleForCloudHSMfunção. 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 mais informações, consulte Excluir um perfil vinculado ao serviço no Guia do usuário do IAM.

Conformidade

Para clusters no modo FIPS, AWS CloudHSM fornece HSMs aprovados pelo FIPS que atendem aos requisitos de conformidade PCI-PIN, PCI-3DS e 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, consulteAWS CloudHSM modos de cluster e tipos de HSM.

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. Os HSMs do tipo hsm1.medium fornecidos pela AWS CloudHSM são certificados FIPS 140-2 nível 3 (Certificado #4218). 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 pela 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 do número de identificação pessoal (PIN), informações usadas para transações em caixas eletrônicos e terminais point-of-sale (POS). Os HSMs hsm1.medium fornecidos pela são compatíveis com PCI PIN AWS CloudHSM desde janeiro de 2023. 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. Os HSMs do tipo hsm1.medium fornecidos pela AWS CloudHSM são compatíveis com PCI-3DS.

SOC 2

O SOC2 é uma estrutura para ajudar organizações de serviços a demonstrar seus controles de segurança na nuvem e no datacenter. O AWS CloudHSM implementou controles de SOC2 em áreas críticas para aderir aos princípios de serviços confiáveis. Para obter mais informações, consulte a página de perguntas frequentes do AWS SOC.

AWS CloudHSM Perguntas frequentes sobre conformidade com PCI-PIN

O PCI PIN fornece requisitos de segurança e padrões de avaliação para transmissão, processamento e gerenciamento de dados do número de identificação pessoal (PIN), informações usadas para transações em caixas eletrônicos e 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.

Perguntas frequentes

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 dos 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 HSMs de uso geral. Com o tempo, podemos fornecer funções de pagamento. Embora o serviço não execute funções de pagamento diretamente, o atestado de conformidade com o PIN AWS CloudHSM PCI permite que os clientes obtenham sua própria conformidade com o PCI para os 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 e SOC2. Esta página lista as alterações que se aplicam atualmente.

Conformidade com o FIPS 140: suspensão do mecanismo de 2024

O Instituto nacional de padrões e tecnologia (National Institute of Standards and Technology, NIST) [1] informa que o suporte à criptografia Triple DES (DESede, 3DES, DES3) e ao encapsulamento e desencapsulamento de chaves RSA com preenchimento PKCS nº 1 v1.5 não será permitido após 31 de dezembro de 2023. Portanto, o suporte para eles termina em 1º de janeiro de 2024 em nossos clusters do modo Federal Information Processing Standard (FIPS). Support para eles permanece para clusters no modo não FIPS.

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 comprimentos 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 aplicativos e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são mais 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 recursos do AWS CloudHSM para oferecer suporte à resiliência, consulte 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 você pode 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 ter mais informações, consulte Crie uma nuvem privada virtual (VPC)..

Ao criar um HSM, AWS CloudHSM coloque uma interface de rede elástica (ENI) em sua sub-rede para que você possa interagir com seus HSMs. Para ter mais informações, consulte Arquitetura do cluster.

AWS CloudHSM cria um grupo de segurança que permite a comunicação de entrada e saída entre HSMs em seu cluster. É possível usar esse grupo de segurança para permitir que as instâncias do EC2 se comuniquem com os HSMs em seu cluster. Para ter mais informações, consulte Configurar os grupos de segurança da instância do cliente Amazon EC2.

Autorização dos usuários

Com AWS CloudHSM, as operações realizadas no HSM exigem as credenciais de um usuário autenticado do HSM. Para ter mais informações, consulte the section called “Noções básicas sobre usuários do HSM”.

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 PrivateLinkuma tecnologia que permite que você acesse AWS CloudHSM APIs de forma privada sem um gateway de internet, dispositivo NAT, conexão VPN ou conexão do AWS Direct Connect. As instâncias na sua VPC não precisam de endereços IP públicos para se comunicar com AWS CloudHSM as 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.

Criar um endpoint da VPC de interface para o 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 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, o nome de host AWS CloudHSM DNS padrão (`https://cloudhsmv2.<region>.amazonaws.com`) será resolvido para seu VPC endpoint.

Essa opção facilita usar o endpoint da VPC. Os AWS SDKs AWS CLI usam 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 executar ações.
- As ações que podem ser executadas.
- Os recursos sobre os quais as ações podem ser realizadas.

Para obter mais informações, consulte Controlar o acesso a serviços com endpoints da VPC no Guia do Usuário do 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 outras AWS CloudHSM ações e suas 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. 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

Problemas conhecidos

AWS CloudHSM tem os seguintes problemas conhecidos. Escolha uma área de tópico para saber mais.

Tópicos

- Problemas conhecidos para a integração de aplicativos de terceiros

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 da chave dos HSMs em seu cluster dentro de um curto período, o cliente entrará em um estado de erro não processado
- 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 cert/chave do arquivo pfx fornecido. Erro: NotPkcs 8"

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 desembrulhar usando esse algoritmo interno AWS CloudHSM. No entanto, as chaves embaladas com AWS CloudHSM não podem ser desempacotadas em outros HSMs ou softwares que esperam 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 cada HSM no cluster e, em seguida, adicionar outro HSM, o que gerará um novo endereço IP, o daemon do cliente continuará pesquisando os HSMs em seus endereços IP originais.
- Solução alternativa: se você executar uma carga de trabalho intermitente, recomendamos usar o `IpAddress` argumento na CreateHsmfunção 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 SDK 5 do cliente falhará explicitamente se o buffer de dados for maior que 64 KB. Você deve atualizar seu cliente e SDK (s) para uma versão maior que 5.0.0 ou superior 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 unWrapKeyartigos 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 da chave dos HSMs em seu cluster dentro de um curto período, o cliente entrará em um estado de erro não processado

- Impacto: quando o cliente encontra o estado de erro não processado, ele congela e deve ser reiniciado.
- Solução alternativa: teste sua taxa de transferência 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 o PKCS11, as chaves HMAC estarão 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 Version 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 obter orientação sobre como fazer isso, consulte Configurar as opções de metadados da instância.

Problema: você recebe o erro “Falha ao criar cert/chave do arquivo pfx fornecido. Erro: NotPkcs 8"

- Impacto: os usuários do SDK 5.11.0 que reconfigurarem o SSL com um certificado e uma chave privada falharão se suas chaves privadas não estiverem no formato PKCS8.
- Solução alternativa: você pode converter a chave privada SSL personalizada para o formato PKCS8 com o comando openssl: `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 SDK do cliente. Você deve atualizar para essa versão do cliente ou posterior para se beneficiar da correção.

Problemas conhecidos para instâncias hsm2m.medium

Os problemas a seguir afetam todas as instâncias hsm2m.medium.

Tópicos

- Problema: a latência de login aumenta devido ao aumento das iterações do PBKDF2
- 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 latência de login aumenta devido ao aumento das iterações do PBKDF2

- Impacto: para aumentar a segurança, o hsm2m.medium executa 60.000 iterações da Função de Derivação de Chave Baseada em Senha 2 (PBKDF2) durante solicitações de login, em comparação com 1.000 no hsm1.medium. Esse aumento pode resultar em uma latência aumentada de até 2 segundos (2s) por solicitação de login.

  O tempo limite padrão para os SDKs do AWS CloudHSM cliente é 20s. As solicitações de login podem expirar e resultar em um erro.
- Solução alternativa: se possível, serialize as solicitações de login no mesmo aplicativo para evitar latência prolongada durante o login.
- Status da resolução: as versões futuras do SDK do cliente terão um tempo limite padrão maior para solicitações de login para compensar esse aumento de latência.

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 isso`User type should be CO or CU`.
- Resolução: versões futuras do SDK do cliente resolverão esse problema. As atualizações serão anunciadas em nosso guia do usuário. Histórico do documento

Problemas conhecidos da biblioteca do PKCS#11

Tópicos

- Problema: o encapsulamentoo de chave AES na versão 3.0.0 da biblioteca PKCS #11 não valida IVs 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 plataformas EL6, como CentOS6 e RHEL 6
- 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 encapsulamentoo de chave AES na versão 3.0.0 da biblioteca PKCS #11 não valida IVs 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 IVs personalizados com comprimento não compatível para o encapsulamento de chaves do AES.
  - 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.

Os IVs 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: vamos corrigir o cliente e os SDKs para implementar corretamente o hash em 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 usando o `CKM_ECDH1_DERIVE` mecanismo está disponível primeiro no cliente e depois importada para o HSM. Um identificador de chave é retornado ao aplicativo.
- Solução alternativa: se você estiver implementando o descarregamento SSL/TLS no AWS CloudHSM, essa limitação pode não ser um problema. Se o aplicativo 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: estamos desenvolvendo a opção para executar inteiramente a derivação de chaves ECDH no HSM. A implementação atualizada será anunciada na página de histórico de versões quando estiver disponível.

Problema: a verificação das assinaturas secp256k1 falha em plataformas EL6, como CentOS6 e RHEL 6

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 Secp256k1 não tem suporte do pacote OpenSSL padrão nas plataformas do EL6, haverá falha na inicialização.

- Impacto: haverá falha na verificação de assinatura do secp256k1 nas plataformas do EL6. Haverá falha na chamada para verificar com um erro `CKR_HOST_MEMORY`.
- Solução alternativa: recomendamos usar o Amazon Linux 1 ou qualquer plataforma do EL7 se o seu aplicativo PKCS #11 precisa verificar assinaturas do 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 descriptografados 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_EncryptUpdate`seguido por `C_Encrypt`
  - `C_DecryptInit`/`C_DecryptUpdate`seguido por `C_Decrypt`
  - `C_SignInit`/`C_SignUpdate`seguido por `C_Sign`
  - `C_VerifyInit`/`C_VerifyUpdate`seguido por `C_Verify`
  - `C_FindObjectsInit`seguido 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 SDK do JCE

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

- Impacto: esse problema pode fazer com que o espaço de chaves dos HSMs se esgote inesperadamente e ocorra quando o 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 aumentam enquanto o aplicativo está em execução e podem fazer com que o espaço livre de chaves dos HSMs se esgote, 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: estamos adicionando suporte ao armazenamento de certificados no repositório de chaves. Esse recurso será anunciado na página de histórico de versões, quando estiver disponível.

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 descriptografia. Como o FIPS requer que o vetor de inicialização (IV) para AES-GCM seja gerado no HSM, o IV de cada parte dos dados criptografados por AES-GCM 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 ao aplicativo.
- Solução alternativa: se você estiver implementando o SSL/TLS Offload in AWS CloudHSM, essa limitação pode não ser um problema. Se o aplicativo 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: estamos desenvolvendo a opção para executar inteiramente a derivação de chaves ECDH no HSM. Quando disponível, anunciaremos a implementação atualizada na página de histórico de versões.

Problema: KeyGenerator e 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 Client SDK 5.4.0 a 5.4.2 esperam incorretamente que o tamanho da chave seja fornecido às APIs especificadas como bytes.
- 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
```
```

Esses avisos não têm impacto. Estamos cientes desse problema e estamos trabalhando para resolvê-lo. Nenhuma resolução nem solução alternativa é necessária.

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
KeyAgreement	`generateSecret()` ou `generateSecret(String)`
KeyPairGenerator	`generateKeyPair()` , `genKeyPair()` , ou `reset()`
KeyStore	Nenhum método é necessário
Mac	`doFinal()` ou `reset()`
MessageDigest	`digest()` ou `reset()`
SecretKeyFactory	Nenhum método é necessário
SecureRandom	Nenhum método é necessário
Assinatura	`sign()` no modo de sinal `verify()` 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 SDK 5 do cliente com operações GetKey

- Impacto: a `getKey` operação da API tem um vazamento de memória no JCE nas versões 5.10.0 e anteriores do SDK do Cliente. Se você estiver usando a `getKey` API várias vezes em seu aplicativo, isso aumentará o crescimento da memória e, consequentemente, aumentará o espaço ocupado pela memória em seu aplicativo. Com o tempo, isso pode causar erros de limitação ou exigir que o aplicativo seja reiniciado.
- Solução alternativa: recomendamos a atualização para o Client SDK 5.11.0. Se isso não puder ser feito, recomendamos não chamar a `getKey` API várias vezes em seu aplicativo. Em vez disso, reutilize a chave retornada anteriormente da `getKey` operação anterior, tanto quanto possível.
- Status da resolução: atualize a versão do SDK do cliente para 5.11.0 ou posterior, o que inclui uma correção para esse problema.

Problemas conhecidos para o OpenSSL Dynamic Engine

Problemas conhecidos para o OpenSSL Dynamic Engine

Tópicos

Problema: você não pode instalar o AWS CloudHSM OpenSSL Dynamic Engine no RHEL 6 e no CentOS6

- 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 descriptografia 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 ou a Provedor JCE.
- 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 em aplicativos 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: suspensão de 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 assinar/verificar assinaturas criptográficas 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+) nas notas de lançamento do 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.

Problemas conhecidos para instâncias do Amazon EC2 que executam o Amazon Linux 2

Problema: o Amazon Linux 2 versão 2018.07 usa um **`ncurses`** pacote atualizado (versão 6) que atualmente é incompatível com os SDKs AWS CloudHSM

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 para a integração de 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

Falhas de sincronização de chaves 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 Client 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 de HSMs especificado nas configurações de sincronização do lado do cliente. Para obter mais informações, consulte Sincronização de chaves no AWS CloudHSM.

Client SDK 3: verifique o desempenho do HSM com a ferramenta pkpspeed

Este tópico descreve como verificar o desempenho do 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 SDK do cliente 3. A ferramenta pkpspeed é executada em condições ideais e chama diretamente o HSM para executar operações sem passar por um SDK como o 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 Instalar e configurar o AWS CloudHSM cliente (Linux). Para obter mais informações sobre como instalar o cliente em uma instância do Windows, consulte Instalar e configurar o AWS CloudHSM cliente (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 as AWS CloudHSM perguntas frequentes para obter detalhes). Isso pode ser verificado usando as ferramentas da CLI, conforme documentado no Guia do AWS CloudHSM usuário, e executando o getHSMInfo 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.
- Basic AES: testa a criptografia AES CBC/ECB.
- 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 para clusters no modo FIPS após 2023. Para clusters no modo não FIPS, 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/...
```

O usuário do Client SDK 5 contém valores inconsistentes

O `user list` comando 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 em

HSMs 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”	`{ "username": "test_user", "role": "inconsistent ", "locked": "false", "mfa": [], "cluster-coverage": "full" }`	Esse usuário está CryptoUser em alguns HSMs e é administrador em outros HSMs. Isso pode acontecer se dois SDKs tentarem criar o mesmo usuário, ao mesmo tempo, com funções diferentes. Você deve remover esse usuário e recriá-lo com a função desejada.	1. Faça login como administrador. 2. Exclua o usuário em todos os HSMs: user delete --username **<user's name>** --role admin user delete --username **<user's name>** --role crypto-user 3. Crie o usuário com a função desejada:

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação
			user create --username **<user's name>** --role **<desired role>**

Propriedade inconsistente	Exemplo de saída da lista de usuários	Implicação	Método de recuperação

A “cobertura do cluster” do usuário é “inconsistente”	{
	"username":
	"test_user",
	
	"role": "crypto-user",
	"locked":
	"false",
	"mfa": [],
	"cluster-coverage":
	"**inconsistent** "
	}
		Esse usuário existe em um subconjunto de HSMs no cluster. Isso pode acontecer se um user create for parcialmente bem-sucedido ou user delete parcialmente bem-sucedido.
		
		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:
			
			user delete --username**<user's name>** --role admin
			3. Execute o seguinte comando:
			
			user delete --username**<user's name>** --role crypto-user
			
			Se o usuário deve existir, siga estas etapas:
			
			1. Faça login como administrador.
			2. Execute o seguinte comando:
			
			user create --username **<user's name>** --role **<desired role>**

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 de HSMs no 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:
			
			user change-mfa token-sign --username **<user's name>** --role **<desired role>** --disable
			
			3. Altere a senha do usuário para que ele possa fazer login em todos os HSMs:
			
			user change-password --username **<user's name>** --role **<desired role>**
			
			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”	{
	"username":
	"test_user",
	
	"role": "crypto-user",
	"locked":
	"false",
	"mfa": [
	{
	"strategy":
	"token-sign",
	"status":
	"**inconsistent** "
	}
	],
	"cluster-coverage":
	"full"
	}
		Esse usuário tem diferentes sinalizadores de MFA em diferentes HSMs no cluster.
		
		Isso pode acontecer se uma operação de MFA for concluída somente 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:
			
			user change-mfa token-sign --username **<user's name>** --role **<desired role>** --disable
			3. Também será necessário alterar a senha do usuário para que ele possa fazer login em todos os HSMs:
			
			user change-password --username **<user's name>** --role **<desired role>**
			
			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): user change-mfa token-sign --username **<user's name>** --role **<desired role>** —token <File>

Erro visto durante a verificação da disponibilidade da chave

Problema: um 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 key generate-symmetric ouchave generate-asymmetric-pair .

- Uma lista de chaves operação foi iniciada.
- Uma nova instância do SDK foi iniciada.

> **Note**
>
> O OpenSSL frequentemente bifurca novas instâncias do SDK.

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 Parâmetros 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.

Extrair 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.

Se `getEncoded`, `getPrivateExponent` e `getS` retornarem nulos após serem ativados, sua chave não atende aos pré-requisitos corretos. Para mais informações, consulte Extração de chaves usando JCE.

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
- Proteger e extrair chaves de um HSM

Controle de utilização do HSM

Quando sua workload exceder a capacidade do HSM do cluster, você receberá mensagens de erro informando que os HSMs estão ocupados ou limitados. Quando isso acontece, você pode ver uma throughput reduzida ou um aumento na taxa de solicitações de rejeição de HSMs. Além disso, os HSMs podem enviar os seguintes erros de ocupação.

Para Client SDK 5

- No PKCS11, os erros de ocupação são mapeados para `CKR_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.`
- Os erros de ocupação de outros SDKs imprimem a seguinte mensagem: `Received error response code from Server. Response Code: 187`.

Para Client SDK 3

- No PKCS11, os erros de ocupação são mapeados para `CKR_OPERATION_ACTIVE`.
- 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.
- Os erros de ocupação de outros SDKs imprimem 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 emAdicionar ou remover 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.

Manter os usuários do HSM em sincronia entre os HSMs no cluster

Para gerenciar os usuários do seu HSM, você usa uma ferramenta de linha de AWS CloudHSM comando conhecida como cloudhsm_mgmt_util. Ela se comunica apenas com os HSMs presentes no arquivo de configuração da ferramenta. Ela não reconhece os outros HSMs no cluster que não estão presentes no arquivo de configuração.

AWS CloudHSM sincroniza as chaves dos seus HSMs em todos os outros HSMs no cluster, mas não sincroniza os usuários ou as políticas do HSM. Ao usar cloudhsm_mgmt_util para gerenciar usuários de HSM, essas alterações do usuário podem afetar apenas alguns dos HSMs do cluster, aqueles presentes no arquivo de configuração cloudhsm_mgmt_util. Isso pode causar problemas ao AWS CloudHSM sincronizar chaves entre HSMs no cluster, porque os usuários que possuem as chaves podem não existir em todos os HSMs do cluster.

Para evitar esses problemas, edite o arquivo de configuração cloudhsm_mgmt_util antes de gerenciar os usuários. Para ter mais informações, consulte ???.

A conexão com o cluster foi perdida

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.

   - Visualize a guia HSMs na página de detalhes do cluster no console do AWS CloudHSM.
   - Use o AWS Command Line Interface (AWS CLI) para emitir o describe-clusterscomando.

   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 tiver criado um cluster antes de 20 de janeiro de 2018, você precisará configurar manualmente uma função vinculada ao serviço para habilitar a entrega dos logs 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.

IVs personalizados com comprimento não compatível para o agrupamento de chaves AES

Este tópico de solução de problemas ajuda a determinar se seu aplicativo gera chaves agrupadas irrecuperáveis. Se você for afetado por esse problema, use este tópico para resolvê-lo.

Tópicos

- Determine se seu código gera chaves agrupadas irrecuperáveis
- Ações que você deve realizar se seu código gerar chaves agrupadas irrecuperáveis

Determine se seu código gera chaves agrupadas irrecuperáveis

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 `libpkcs11.so` na sua pasta `/opt/cloudhsm/lib` . Aplicativos escritos na linguagem C geralmente usam a bibliotec

Condição	Como posso saber?
	a PKCS #11 diretamente, enquanto aplicativos 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.
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: `rpm -qa | grep ^cloudhsm`
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 agrupamento baseados em AES que usam initialization vectors (IVs, vetores de inicialização), como `CKM_AES_GCM` e `CKM_CLOUDHSM_AES_GCM` , não são afetados por esse problema. Saiba mais sobre funções e mecanismos.

Condição	Como posso saber?
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: `CK_MECHANISM mech = {CKM_AES_KEY_WRAP, IV_POINTER, IV_LENGTH};` 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 irrecuperáveis”.

Ações que você deve realizar se seu código gerar chaves agrupadas irrecuperáveis

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 desabilitaremos explicitamente IVs personalizados para agrupamento 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 agrupadas usando a versão 3.0.0 da biblioteca PKCS #11 e, em seguida, entrar em contato com o suporte (https://aws.amazon.com/support) para obter assistência 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.

Resolver falhas na criação do cluster

Quando você cria um cluster, AWS CloudHSM cria a função AWSServiceRoleForCloudHSM vinculada ao serviço, 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 AWS Support. Para obter mais informações sobre a função AWSServiceRoleForCloudHSM vinculada ao serviço, 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 AWSServiceRoleForCloudHSM serviço. 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 Perfis. 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",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "cloudhsm.amazonaws.com"
                    }
                }
            ]
        },
        "RoleId": "AROAJ4I6WN5QVGG5G7CBY",
        "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"
    }
}
```

Recuperação de logs de configuração 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

- Ferramenta de suporte do Client SDK 5
- Ferramenta de suporte do Client SDK 3

Ferramenta de suporte do Client SDK 5

O script 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 ]_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`.

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.

  ```
  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.

```
C:\Program Files\Amazon\CloudHSM\bin\jce_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` Depois de configurado para`term`, o ambiente sem servidor poderá produzir para. CloudWatch

Para obter os registros do cliente CloudWatch, consulte Trabalho com grupos e fluxos de registros no Guia do usuário do Amazon CloudWatch Logs.

Ferramenta de suporte do Client SDK 3

O script 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 de Cotas de serviço. No console, escolha seu serviço e cota e envie sua solicitação. Para obter mais informações, consulte a documentação das Cotas de serviço.

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 teclas, com chaves assimétricas com um máximo de 3.333
Máximo de usuários por cluster	1,024	1,024
Comprimento máximo do nome do usuário	31 caracteres	31 caracteres
Comprimento obrigatório da senha	8 a 32 caracteres	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.

Para ter mais informações, consulte Recursos do sistema.

Recursos do sistema

As cotas de recursos do sistema são cotas sobre o que o AWS CloudHSM cliente pode usar durante a execução.

Os descritores de arquivos são um mecanismo do sistema operacional para identificar e gerenciar os arquivos abertos por processo.

O daemon cliente do CloudHSM utiliza descritores de arquivos para gerenciar conexões entre os aplicativos e o cliente, bem como entre o cliente e o servidor.

Por padrão, a configuração do cliente do CloudHSM alocará 3.000 descritores de arquivos. Esse valor padrão foi projetado para gerar uma capacidade de sessão e threading ideal entre o daemon do cliente e seus HSMs.

Em raras circunstâncias, se você estiver executando seu cliente em um ambiente de recursos restritos, poderá ser necessário alterar esses valores padrão.

> **Note**
>
> Ao alterar esses valores, o desempenho do cliente do CloudHSM pode sofrer e/ou o aplicativo pode se tornar inoperável.

1. Edite o arquivo `/etc/security/limits.d/cloudhsm.conf`.

```
#
# DO NOT EDIT THIS FILE
#
hsmuser soft nofile 3000
hsmuser hard nofile 3000
```

2. Edite os valores numéricos, conforme necessário.

> **Note**
>
> A cota `soft` deve ser inferior ou igual à cota `hard`.

3. Reinicie o processo do daemon do cliente do CloudHSM.

> **Note**
>
> Essa opção de configuração não está disponível nas plataformas do Microsoft Windows.

Downloads para o SDK AWS CloudHSM do cliente

Downloads

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, com exceção do suporte para fornecedores de GNV e KSP. Para ter mais informações, consulte Comparação de componentes do Client SDK.

> **Note**
>
> Para obter informações sobre quais plataformas são suportadas por cada SDK do cliente, consulte Plataformas compatíveis com o Client SDK 5 e. Plataformas compatíveis com o Client SDK 3

Versão mais recente

Esta seção inclui a versão mais recente do SDK do cliente.

Versão 5 do SDK do cliente: versão 5.12.0

Amazon Linux 2

Baixe o software versão 5.12.0 para Amazon Linux 2 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 383baed4a861391eb0923c0d9cf451851c6dd02d7d6a9e9cc3638c60bf300ef2)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum f7aba68787a4c975f3e9f4ead28c2c28adc787ca0babebc070a928d226ff330a)
- Provedor JCE (SHA256 checksum 1f75f1a5d428b18ce2dc6ce8e17923009895c2545e2d04d76dafd6da914c0b4e)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum 4c27fae1ef5fd1642c04514ec84ad4cab78f59a32eb3fce59b51805c44b25295)

Baixe o software versão 5.12.0 para Amazon Linux 2 na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum c28a1f27e23e6ab1550dab6a353c6c9338a391a84d57f4ac99a1a3a9810c753f)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 7d2e864c31c13f55443c1b1d04589fbdd4558fe103954de4384691e2c429a872)
- Provedor JCE (SHA256 checksum e9a35eb87b2f257c47fb083d286deb835da45858b2d89759ca7d5bb4ef747b4b)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum 28b6f918912b5c63bf10018824b642a805b309c21947a1d0ebbdc44647e80554)

Amazon Linux 2023

Baixe o software versão 5.12.0 para Amazon Linux 2023 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 02801365cba449c5238a4e5ad3df1ddf7edd00ade976f47e956e885286503f3f)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 0abed69a7c6acaafdaabdcc5fab7d56611ffd94f5480cade6f8beace9aeae056)
- Provedor JCE (SHA256 checksum 3d5d9a903d3a216eca40f92dbb0b4030b7a86ad7ceee8d62241c97a6e1881e25)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum f96671d882b862033bba0b3633448dc6a26e45a25063e29b79a5cd4b7fc4945c)

Baixe o software versão 5.12.0 para Amazon Linux 2023 na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum 53d05006b46bda8e9c1dd76e8307a780bfe0a67b10a9a87723c97f94e29f5b8e)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum ec1cca8e01b3303ff9473eeef6b33dc85b6affac7a47387b098905f9f2fc85ba)
- Provedor JCE (SHA256 checksum c828ae56f46233215b9f35798b5859ebdac962af442acbc457081c3baaa44f11)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum ddd5dcd68d01f4fafaf13dc0b4ddcf98e3731ed51bdd51f85535b29353644a9f)

CentOS 7 (7.8+)

Baixe o software versão 5.12.0 para CentOS 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 383baed4a861391eb0923c0d9cf451851c6dd02d7d6a9e9cc3638c60bf300ef2)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum f7aba68787a4c975f3e9f4ead28c2c28adc787ca0babebc070a928d226ff330a)
- Provedor JCE (SHA256 checksum 1f75f1a5d428b18ce2dc6ce8e17923009895c2545e2d04d76dafd6da914c0b4e)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum 4c27fae1ef5fd1642c04514ec84ad4cab78f59a32eb3fce59b51805c44b25295)

RHEL 7 (7.8+)

Baixe o software da versão 5.12.0 para RHEL 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 383baed4a861391eb0923c0d9cf451851c6dd02d7d6a9e9cc3638c60bf300ef2)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum f7aba68787a4c975f3e9f4ead28c2c28adc787ca0babebc070a928d226ff330a)
- Provedor JCE (SHA256 checksum 1f75f1a5d428b18ce2dc6ce8e17923009895c2545e2d04d76dafd6da914c0b4e)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum 4c27fae1ef5fd1642c04514ec84ad4cab78f59a32eb3fce59b51805c44b25295)

RHEL 8 (8.3+)

Baixe o software da versão 5.12.0 para RHEL 8 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 6e51e95122fd0991278888287f0c408808b26fb5f1196c46168477b9090fc478)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 1f1d52ff7af6c537d8cfeb5973c691a9d90a518accd685ff9b66cd78daf98928)
- Provedor JCE (SHA256 checksum 156944607de987d6b39bd8a2d21ccd294c01377a9e35f9f15f8b0f4c8bb90033)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum 351e802f79dd2d0b5f7d23bb74c146be05e5169b603c9aace24189094a45a35d)

RHEL 9 (9.2+)

Baixe o software da versão 5.12.0 para RHEL 9 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum d1b2f4ac7e6e0c18e788512e7726bc68b571d99a1442ce2f2e80f4b0f9956266)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum cf86a3f17cd6c51969d4ce80c1e3ea6513b995611be7e2e72e5e5233c71d6add)
- Provedor JCE (SHA256 checksum ae89e256eb89ec6b4fa0f001e7a4e1d8f1c08530423e81aa74d69a17b25d9a99)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum dfe6fe5d890c33b2f5d38f906ade113b06c8c05f3427a327744c454e7302f1a5)

Baixe o software versão 5.12.0 para RHEL 9 na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum cad72a6ab2232b4c38b90d7c62147520b975d646773dd90d7be897fa0a537d2d)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum ad751f756530a2317c3c64380ea3a07865b13e1874fab0e61ac530b21487c7fb)
- Provedor JCE (SHA256 checksum d204e69acfbb90996fb08ae3573607b65630b1124fb379e078c002d55ac07766)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum c0f412cc59bafd235e046cdc1a0c5d330f2d72f7d6434672e9522f86bc945090)

Ubuntu 20.04 LTS

Baixe o software versão 5.12.0 para o Ubuntu 20.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum d37b1f872eb2b1ab34303d5b8b803daa925902b645c57c6e15a28bb6321e0f42)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum cdc6e737652556b57d26d8816b2bc9820128cb3919360660b6f7fe65f9d39e3f)
- Provedor JCE (SHA256 checksum f567a08344414a4776e1c5a9715657476925ca32695c4c2dd84a4f3fc5dc1615)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum f2ee5ad01c5018fc3670f602228fd71087228cd3923bf5b9bc73e4d7084dac6c)

Ubuntu 22.04 LTS

Baixe o software versão 5.12.0 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 0e78928acd7a1662e4b07b15d5c3ccb88714ff89e47b991c8ab6e4c2229ee5aa)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 4f3168745edc5592234891a7b1d82b179a4947e87c72fade1be3bad58b7ed1a3)
- Provedor JCE (SHA256 checksum d4c3655cdc2b00d1ab5ceafac94dfbc5c5244ed20e10fdd9db9f4e741e013733)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum d00bbacb6f2e57bd92d832a2bd11cadede972f8e82cc402ec0684b9c6b23123c)

Baixe o software versão 5.12.0 para o Ubuntu 22.04 LTS na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum 0c1121535c523acb864215338292bab32acee438357878b5fc0b6d268713b86f)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum dc7a219302021570bc8c36674d2bd33165557bb2f9a0af8fdf114f1b85a70d84)
- Provedor JCE (SHA256 checksum af3834a10081f1e4e7894275c8b9c7b7649b8de3b6f0aeb0781a3358183a9046)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum baa253ac62c2fbcc5712561e0fb0feb25461efc3ce68cf86d4c7bf0af0f14a34)

Windows Server 2016

Baixe o software versão 5.12.0 para Windows Server 2016 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 11c3255fcc90b47810cfe4b2f71d56a006d295efccdd90f0d3f2dec5d2bab893)
- Provedor JCE (SHA256 checksum 09001458196590f54352c0c8986f442003bfc2db71bac6392ce512899d386806)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum b446ad1387fe406dcc0a12b6de86fa98e9db4a18f9829b745efb87750c6e31ea)

Windows Server 2019

Baixe o software versão 5.12.0 para Windows Server 2019 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 11c3255fcc90b47810cfe4b2f71d56a006d295efccdd90f0d3f2dec5d2bab893)

- Provedor JCE (SHA256 checksum 09001458196590f54352c0c8986f442003bfc2db71bac6392ce512899d386806)
  - Javadocs para AWS CloudHSM (SHA256 checksum 7158bc80e3b5b0915d83c39d4c060060a43a79cc407b1f783383b9e20bc5ff43)
- CLI do CloudHSM (SHA256 checksum b446ad1387fe406dcc0a12b6de86fa98e9db4a18f9829b745efb87750c6e31ea)

O Client SDK 5.12.0 adiciona suporte ARM a várias plataformas e melhorias de desempenho para todos os SDKs. Novos recursos foram adicionados ao provedor de CLI e JCE do CloudHSM.

Suporte à plataforma

- Foi adicionado suporte para Amazon Linux 2023 na arquitetura ARM64 para todos os SDKs.
- Foi adicionado suporte para o Red Hat Enterprise Linux 9 (9.2+) na arquitetura ARM64 para todos os SDKs.
- Foi adicionado suporte para o Ubuntu 22.04 LTS na arquitetura ARM64 para todos os SDKs.

CLI do CloudHSM

- Foi adicionado o seguinte comando:
  - replicação de chave
- Foi adicionado suporte para conexão com vários clusters. Para ter mais informações, consulte Conectando-se a vários clusters com o CloudHSM CLI.

Provedor JCE

- Adicionado `KeyReferenceSpec` para recuperar chaves usando`KeyStoreWithAttributes`.
- Adicionado `getKeys` para recuperar várias chaves ao mesmo tempo usando`KeyStoreWithAttributes`.

Melhorias de performance

- Melhorias no desempenho da NoPadding operação AES CBC para todos os SDKs.

Versões anteriores do Client SDK

Esta seção lista as versões anteriores do SDK do cliente.

Versão 5.11.0

Amazon Linux 2

Baixe o software versão 5.11.0 para Amazon Linux 2 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 9fc0cd7cf003a7cb7e42dbd19671d58a97fc3b3d871d284dc6ae7fd226598772)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 1df6669c971440d446890b0fbeb74125a423df7b14e7ac4577347be7ef176572)
- Provedor JCE (SHA256 checksum 148a3f1de55a68e3bb525fb2994645333a52c2e9e46946dd8d90fcbc90ab64fd)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum a68f4a56d4c539cfcc8a1e56e19b5ff385bb24936ea5f349255b4e9bfbee9aab)

Baixe o software versão 5.11.0 para Amazon Linux 2 na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum 5ac16449ec149c9b5e7776865803245ab17d0f1ad56df80173840c5e8d257b19)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 28c2eb7f3f60172b0186e5c25f71bb7341537058a71f288673936766048083c1)
- Provedor JCE (SHA256 checksum 06c9d9d281c12b1d2bd9a7b601d6317e46cedf175706bbfa3e4dcaed6ba05448)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 218982bb17aa751969a7866b0a9ff27e7aa5007a07817627d9cc1f7d60a78160)

Amazon Linux 2023

Baixe o software versão 5.11.0 para Amazon Linux 2023 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 55310ab333d18bcfabdc4b74115b040386b4508934bdff93e1d054c4c4a6f9ea)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum f3d4934dc872a9b5212a180b9814ca2af3eca01ee228a8725563f1770add0dce)
- Provedor JCE (SHA256 checksum 757d3abb515aeb08f4b1c83970ee0979399efee00ee78c9a9dbec05f4ed9768d)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 22af8f0501ff9a45a9e0683a408a63771c2c06c66abf5478d310d6d32e013555)

CentOS 7 (7.8+)

Baixe o software versão 5.11.0 para CentOS 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 9fc0cd7cf003a7cb7e42dbd19671d58a97fc3b3d871d284dc6ae7fd226598772)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 1df6669c971440d446890b0fbeb74125a423df7b14e7ac4577347be7ef176572)
- Provedor JCE (SHA256 checksum 148a3f1de55a68e3bb525fb2994645333a52c2e9e46946dd8d90fcbc90ab64fd)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum a68f4a56d4c539cfcc8a1e56e19b5ff385bb24936ea5f349255b4e9bfbee9aab)

RHEL 7 (7.8+)

Baixe o software da versão 5.11.0 para RHEL 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 9fc0cd7cf003a7cb7e42dbd19671d58a97fc3b3d871d284dc6ae7fd226598772)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 1df6669c971440d446890b0fbeb74125a423df7b14e7ac4577347be7ef176572)
- Provedor JCE (SHA256 checksum 148a3f1de55a68e3bb525fb2994645333a52c2e9e46946dd8d90fcbc90ab64fd)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum a68f4a56d4c539cfcc8a1e56e19b5ff385bb24936ea5f349255b4e9bfbee9aab)

RHEL 8 (8.3+)

Baixe o software da versão 5.11.0 para RHEL 8 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum b95b9f588656fb14fd08bb66ce0e0da807b96daa38348dec07a508c9bef7403a)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 7bb437b91a52e863b2b00ff7f427ce22522026daf757be873ee031ec6ffffd88)
- Provedor JCE (SHA256 checksum e0db887e05eb535314f4d99f21da12d87d35ebb8baf9726f4ce8f01d9df0ea01)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 8485b5a6d679767ca9b4f611718159a643cf3e85090a8e4d20fe53c3707e25c3)

RHEL 9 (9.2+)

Baixe o software da versão 5.11.0 para RHEL 9 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 87b56a20accf67df53a203b7f115655b2acfaec4516682d4976d9475b10bec8e)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 83a6b58572e985df937beede4b10e867b0ac6050ace8010dc8d535be365d2747)
- Provedor JCE (SHA256 checksum ee95213d02d913250478d0793d6dd578e5c54d765e635c7468a49bdff4c2a6f3)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 7e168ed3bef8e9c5110645e9960680e9a57f7b94e16aec71422e3c67ebc58fb5)

Ubuntu 20.04 LTS

Baixe o software versão 5.11.0 para o Ubuntu 20.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum abc3a339d1fe5850db65620804e9a910f8b4f913624ef9b7189f2f0df1825c01)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 075fc3f9974d552f27ad67fa92c8abff31b756b9add875b8cd4957e6801583a4)
- Provedor JCE (SHA256 checksum 5de45c519133a0dae8da3ac01809db7974be25c14c15eb773fc5c972c0178c13)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 83e0e4505a063792c19feb3d4cfd032b9089091916168d92b0f51a967a007734)

Ubuntu 22.04 LTS

Baixe o software versão 5.11.0 para o Ubuntu 22.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum b8f20be125c8530b2a7bd945956e9c04296fba5634af408b40be4e03bdbad72a)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum d728c156eb4ee5c67159e57d6b092785800baa5fb61c14d64f460a8b8f53a778)
- Provedor JCE (SHA256 checksum 44e943b8cd1176ad666e249342687744a280c6222df58b5a9f084c932f628284)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)

- CLI do CloudHSM (SHA256 checksum 8ccf5389d459611be813e42d7f9d040090f94f3fe88f9d110bcfb25e9619e4a7)

Windows Server 2016

Baixe o software versão 5.11.0 para Windows Server 2016 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum aa4bce5be15bbe0978b7205c619bb91c55a8e0f1f4636be311f24878f7709e07)
- Provedor JCE (SHA256 checksum 004cdb9ecb4a4d72458084997de7f562fb76a4e2f0567009f1dfafa7b2bded47)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 679795db759fda4823232142297a281e21a7d6f32cb5ddd6ac4c479866fa33b7)

Windows Server 2019

Baixe o software versão 5.11.0 para Windows Server 2019 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum aa4bce5be15bbe0978b7205c619bb91c55a8e0f1f4636be311f24878f7709e07)
- Provedor JCE (SHA256 checksum 004cdb9ecb4a4d72458084997de7f562fb76a4e2f0567009f1dfafa7b2bded47)
  - Javadocs para AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI do CloudHSM (SHA256 checksum 679795db759fda4823232142297a281e21a7d6f32cb5ddd6ac4c479866fa33b7)

O Client SDK 5.11.0 adiciona novos recursos, melhora a estabilidade e inclui correções de erros para todos os SDKs.

Suporte à plataforma

- Foi adicionado suporte para Amazon Linux 2023 e RHEL 9 (9.2+) para todos os SDKs.
- O suporte para o Ubuntu 18.04 LTS foi removido devido ao seu recente fim de vida útil.
- O suporte para Amazon Linux foi removido devido ao seu recente fim de vida útil.

CLI do CloudHSM

- Foram adicionados os seguintes comandos:
  - sinal criptográfico
  - verificação criptográfica

  - pem de importação de chaves
  - chave: desembrulhar
  - envoltório de chaves
- key generate-fileagora suporta 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

- Foi adicionado suporte para JDK 17 e JDK 21.
- Foi adicionado suporte para chaves AES a serem usadas em operações HMAC.
- Foi adicionado o novo atributo chave`ID`.
- Introduziu uma nova `DataExceptionCause` variante para esgotamento de chaves:. `DataExceptionCause.KEY_EXHAUSTED`

Correções de bugs e melhorias:

- Aumentamos o tamanho máximo do `label` atributo de 126 para 127 caracteres.
- Corrigido um bug que impedia o desempacotamento das chaves EC com o RsaOaep mecanismo.
- Resolveu um problema conhecido da operação GetKey no provedor JCE. Para mais detalhes, consulte Problema: vazamento de memória do SDK 5 do cliente com operações GetKey.
- Registro aprimorado em todos os SDKs para chaves Triple DES que atingiram seu limite máximo de bloqueio de criptografia, de acordo com o FIPS 140-2.
- Foram adicionados problemas conhecidos para o OpenSSL Dynamic Engine. Para mais detalhes, consulte Problemas conhecidos para o OpenSSL Dynamic Engine.

Versão 5.10.0

Amazon Linux

Faça download do software versão 5.10.0 para Amazon Linux na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum d63adf3e96c19c2d894b2defcbadd916dbb0398993050b1358bd93a36aa5acab)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 4daa3e591ffd5f7ce8ef3759c41deaa38867f5e5d21f15927aea83afb1678ac5)
- Provedor JCE (SHA256 checksum 6c1ac94d3080f1c609d9dafbcb14480911beef3a488c4ed6f2b11b377da9b477)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum c12617fcd7990ba53e96f477979b410e3a5f17842ca7a912861b8b820809b5b5)

Amazon Linux 2

Faça download do software versão 5.10.0 para o Amazon Linux 2 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum fc47e705e57a0bfd433f7b46c9477a70df5c442a8ad9c2969bcef38e328e4933)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 0aca262df6780995c9b884fcb8765bbd64acaf21b2286ec4d05a9a90edb3d4cb)
- Provedor JCE (SHA256 checksum b5be7f73c4bcffc5da6f89f324e6b3db5b091610464c8bd38dbddfff0484b2c2)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum e8cf09966890b88a61e695dc034874a445093300359d5d6a86b5a546803920bb)

Faça download do software versão 5.10.0 para o Amazon Linux 2 na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum 5d8dfd835f1ed5a7f5a4fcc8ecf81cfa29883aca7e2985de69b5db723ab663db)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 91fb8efe2646bf0dbd9087554baa09554714e9d56e9bfd5c0dc3023a9f485574)
- Provedor JCE (SHA256 checksum 99f6e55c37fdf00085a816d46835aeff54470797b3b71f4d28a70dc79c9caf44)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum 4a88ba9b4cf0dd5573f3dd88ab9dc257e4c486069cb529c5d554979ee2dd83af)

CentOS 7 (7.8+)

Faça download do software versão 5.10.0 para CentOS 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum fc47e705e57a0bfd433f7b46c9477a70df5c442a8ad9c2969bcef38e328e4933)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 0aca262df6780995c9b884fcb8765bbd64acaf21b2286ec4d05a9a90edb3d4cb)
- Provedor JCE (SHA256 checksum b5be7f73c4bcffc5da6f89f324e6b3db5b091610464c8bd38dbddfff0484b2c2)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum e8cf09966890b88a61e695dc034874a445093300359d5d6a86b5a546803920bb)

RHEL 7 (7.8+)

Faça download do software versão 5.10.0 para RHEL 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum fc47e705e57a0bfd433f7b46c9477a70df5c442a8ad9c2969bcef38e328e4933)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 0aca262df6780995c9b884fcb8765bbd64acaf21b2286ec4d05a9a90edb3d4cb)
- Provedor JCE (SHA256 checksum b5be7f73c4bcffc5da6f89f324e6b3db5b091610464c8bd38dbddfff0484b2c2)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum e8cf09966890b88a61e695dc034874a445093300359d5d6a86b5a546803920bb)

RHEL 8 (8.3+)

Faça download do software versão 5.10.0 para RHEL 8 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 96afb7042a148ddc7a60ab6235b49e176d0460d1c2957bd76ca3d8406ac1cb03)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 2caad2bffea8aef73c91ad422d09772ef830fe7f80a7be19020e6a107eadfbe8)
- Provedor JCE (SHA256 checksum 3543551f08fbe3900821ea2d4ea148b4e86e2334bc94d7ffef6f3b831457cd71)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum 812eccaadfc490f13bcd0b0a835ef58f3a3d4344ad7e0a237de476dd24509525)

Ubuntu 18.04 LTS

Faça download do software versão 5.10.0 para Ubuntu 18.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum be4c61766b8b46e1f6c14c3dcf90aaab9f38240fcd9c68b4009704276c5f6f4a)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 64bd8af827b6dc3786e8ad28858cbc4ef6a0fd42164a0945f427eddcf5f02858)
- Provedor JCE (SHA256 checksum 9fcbdf08e93641468588b608173f26f18781bbc029ed95b2e086da29a968cc00)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum 13808bddddb7eedeb2b8486d23a9976c7fa8d9220149a6b9400626bcaff3b513)

> **Note**
>
> Devido ao recente fim da vida útil do Ubuntu 18.04 LTS, não AWS CloudHSM será mais possível oferecer suporte a essa plataforma na próxima versão.

Ubuntu 20.04 LTS

Faça download do software versão 5.10.0 para Ubuntu 20.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 99ae96504580ff85ed4958a582903a847f666bdaafafbe887a5a76db58f24500)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 13e3f6fe086acf9617b163f66e3941f973daa583fb9322d16c396aa29fc3611d)
- Provedor JCE (SHA256 checksum 44562cebd9af1aa965840cd9bcb237e518d24c715b3c8bca1405c9c1871835e2)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum ab71b4ec531c5e6d05c91539c7edc1c07e6c748052ebf6200f148cb6812538c5)

Ubuntu 22.04 LTS

Faça download do software versão 5.10.0 para Ubuntu 22.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum ee331a44fbe4936ec98a3ae55d58e67ed38e8bbff0a4f4ce8b1bd8239b75877b)
- O suporte para o OpenSSL Dynamic Engine ainda não está disponível para esta plataforma.

- Provedor JCE (SHA256 checksum 9e44d14dd33624f6fe36711633013e47e4a93f4d4635e08900546113ded56e3d)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum 2df361546848cd3f8965b1007dca42a0c959eb10d9e3f4995e8e1c852406751d)

Windows Server 2016

Faça download do software versão 5.10.0 para Windows Server 2016 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 7aae9bfd99a6dd0f4d376c227c206c01847f83a9efd774d1063d76cc6fdaa89f)
- Provedor JCE (SHA256 checksum 1c58fd651e51be2ba59051a87aceca0452990b29837b8a7efabcd510ccbf8c1f)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum f745a2236c9eb9f6f128313eddc35795bd5e47fdf67332bedeb2554201b61a24)

Windows Server 2019

Faça download do software versão 5.10.0 para Windows Server 2019 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 7aae9bfd99a6dd0f4d376c227c206c01847f83a9efd774d1063d76cc6fdaa89f)
- Provedor JCE (SHA256 checksum 1c58fd651e51be2ba59051a87aceca0452990b29837b8a7efabcd510ccbf8c1f)
  - Javadocs para AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI do CloudHSM (SHA256 checksum f745a2236c9eb9f6f128313eddc35795bd5e47fdf67332bedeb2554201b61a24)

O Client SDK 5.10.0 melhora a estabilidade e inclui correções de erros para todos os SDKs.

CLI do CloudHSM

- Adicionados novos comandos que permitem aos clientes gerenciar chaves usando a CLI do CloudHSM, 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, desagrupar, derivar e criar chaves de sessão para todos os SDKs.
- Corrigido um bug no provedor JCE que impedia que arquivos temporários fossem removidos na saída.
- Corrigido um erro que causava um erro de conexão sob certas condições após a substituição dos HSMs no 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 CLI do CloudHSM (o suporte para OpenSSL Dynamic Engine ainda não está disponível).

Versão 5.9.0

Amazon Linux

Faça download do software versão 5.9.0 para Amazon Linux na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 4f368be41f006b751ac41b14e1435c27841f60bbde0f032ec02a359fea637dcf)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 81af0d34683825cd6ff844ccacf9c8f4842a4ba76e3875a89121d09a286b4490)
- Provedor JCE (SHA256 checksum e8e5bc09d8e0b3cb24f30ab420fe08902a19073012335ac94382ec55fcc45abd)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 17284144b45043204ce012fe8b62b1973f10068950abedbd9c2c6172ed0979c6)

Amazon Linux 2

Faça download do software versão 5.9.0 para o Amazon Linux 2 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum e5affca37abc4ff76369237649830feb32fccd3fa05199cc2021230137093c56)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 848a2e31550bbc2b0223468877baa2a8cda3131ef8537856b31db226d55c4170)
- Provedor JCE (SHA256 checksum 884f483ef3e9c7def92e3ff01b226e5cbf276d96dcb2f6f56009516f19d41dc0)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 2e62d5a27cff46d9fb47d656afeccd9dbfb5413bfd2267dd3c8fb7960fef7f26)

Faça download do software versão 5.9.0 para o Amazon Linux 2 na arquitetura ARM64:

- Biblioteca PKCS #11 (SHA256 checksum 4337dca5a08c5194b1118fa197bb4a4f7988df4e1b961e6f2e367295ba99d61d)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 4f08689934e877662a7ce64554fb04eb4b2c213b936018609ff187d100e34a85)
- Provedor JCE (SHA256 checksum b337b80271a2d308949d5911971fe6ad35df4e34876a481fcac347f1d897fe39)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum a4d466e6b5f74dcd283ba32c9dd87441941d5e5a05936b7c2b4cc7ef85eb1071)

CentOS 7 (7.8+)

Faça download do software versão 5.9.0 para CentOS 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum e5affca37abc4ff76369237649830feb32fccd3fa05199cc2021230137093c56)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 848a2e31550bbc2b0223468877baa2a8cda3131ef8537856b31db226d55c4170)
- Provedor JCE (SHA256 checksum 884f483ef3e9c7def92e3ff01b226e5cbf276d96dcb2f6f56009516f19d41dc0)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 2e62d5a27cff46d9fb47d656afeccd9dbfb5413bfd2267dd3c8fb7960fef7f26)

RHEL 7 (7.8+)

Faça download do software versão 5.9.0 para RHEL 7 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum e5affca37abc4ff76369237649830feb32fccd3fa05199cc2021230137093c56)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 848a2e31550bbc2b0223468877baa2a8cda3131ef8537856b31db226d55c4170)
- Provedor JCE (SHA256 checksum 884f483ef3e9c7def92e3ff01b226e5cbf276d96dcb2f6f56009516f19d41dc0)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 2e62d5a27cff46d9fb47d656afeccd9dbfb5413bfd2267dd3c8fb7960fef7f26)

RHEL 8 (8.3+)

Faça download do software versão 5.9.0 para RHEL 8 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 081887f6ea1d9df9d1e409b2b5bde83e965c42229acbeb1f950c8fe478361edc)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 6b0500a42fd57c39f076f14e5079f80145b6ebd2c441395761eb04600c07bda5)
- Provedor JCE (SHA256 checksum 2bc7ac26b259af92a65fbd5a30d5eb2a92ce0e70efe41feb53bf82f168aa90bb)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 79ecbe9b4c5316ccf447d8c59b76b5ac2cc854bd79cd50c1f29197aa8cb080db)

Ubuntu 18.04 LTS

Faça download do software versão 5.9.0 para Ubuntu 18.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum bc6d2227edd7b5a83fed32741fbacbb1756d5df89ebb3435d96f0609a180db65)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 2d6a26434fa6faf337f1dfb42de033220fa405a82d4540e279639a03b3ee6e9d)
- Provedor JCE (SHA256 checksum e12aef122f490e9026452ce31c25625b1accb9a5866b3d470488f10f047f1873)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum f0bcabe594db3e8ff86cc0f65c2a10858d34452eb6b9fc33d7aac05c0f5f4f30)

Ubuntu 20.04 LTS

Faça download do software versão 5.9.0 para Ubuntu 20.04 LTS na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum 15dde8182f432de9e7d369b05e384e1f2d80dcca85db3b16ecc26cdef1a34bb9)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum c8ba94a999038af87d4905b7c1feb4cc87e20d1776a32ef6f6d11ee000b5a896)
- Provedor JCE (SHA256 checksum de33cd3e8130a06d9da5207079533aac8276a1319ac435a3737b4f65bd8fb972)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum cfa31535ad9a99a5113496c06fbace38e9593491aca9bb031a18b51075973e68)

Windows Server 2016

Faça download do software versão 5.9.0 para Windows Server 2016 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum ab5380805b0e17dd89dbbefd3fbda8b54da3c140f82e9f3d021850c31837bbe3)
- Provedor JCE (SHA256 checksum f0941d7a20193818133de8a742d3b848ea19abaf25f5a71ac65949ce5a37c533)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 131530ffe5caff963d483f440d06dcfb41dc11b0f8d78f1dd07bb07f76aeb6d2)

Windows Server 2019

Faça download do software versão 5.9.0 para Windows Server 2019 na arquitetura x86_64:

- Biblioteca PKCS #11 (SHA256 checksum ab5380805b0e17dd89dbbefd3fbda8b54da3c140f82e9f3d021850c31837bbe3)
- Provedor JCE (SHA256 checksum f0941d7a20193818133de8a742d3b848ea19abaf25f5a71ac65949ce5a37c533)
  - Javadocs para AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI do CloudHSM (SHA256 checksum 131530ffe5caff963d483f440d06dcfb41dc11b0f8d78f1dd07bb07f76aeb6d2)

O Client SDK 5.9.0 melhora a estabilidade e inclui correções de erros para todos os SDKs. Foi realizada uma otimização para todos os SDKs para informar os aplicativos sobre falhas imediatas de operação 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

Versão 3.4.4

Para atualizar o Client SDK 3 nas plataformas Linux, você deve usar um cmando em lote que atualiza o daemon do cliente e todas as bibliotecas ao mesmo tempo. Para obter mais informações sobre o upgrade, consulte Atualização do Client SDK 3.

> **Note**
>
> O Client SDK 3 e suas ferramentas de linha de comando relacionadas (Key Management Utility e CloudHSM Management Utility) estão disponíveis somente no tipo HSM hsm1.medium. Para mais detalhes, consulte AWS CloudHSM modos de cluster e tipos de HSM.

Para fazer download do software, escolha a guia de seu sistema operacional preferido e depois o link para cada pacote de software.

Amazon Linux

Faça download do software versão 3.4.4 para o Amazon Linux:

- AWS CloudHSM Cliente (SHA256 checksum 900de424d70f41e661aa636f256a6a79cc43bea6b0fe6eb95c2aaa63e5289505)
- Biblioteca PKCS #11 (SHA256 checksum a3f93f084d59fee5d7c859292bc02cb7e7f15fb06e971171ebf9b52bbd229c30)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 8db07b9843d49016b0b6fec46d39881d94e426fcaae1cee2747be14af9313bb0)
- Provedor JCE (SHA256 checksum 360617c55bf4caa8e6e78ede079ca68cf9ef11473e7918154c22ba908a219843)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum c9961ffe38921131bd6f3702e10d73588e68b8ab10fbb241723e676f4fa8c4fa)

Amazon Linux 2

Faça download do software versão 3.4.4 para o Amazon Linux 2:

- AWS CloudHSM Cliente (SHA256 checksum 7d61d835ae38c6ce121d102b516527f342a76ac31733768097d5cab8bc482610)
- Biblioteca PKCS #11 (SHA256 checksum 2099f324ff625e1a46d96c1d5084263ca1d650424d7465ead43fe767d6687f36)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 6d8e81ad1208652904fe4b6abc4f174e866303f2302a6551c3fbef617337e663)
- Provedor JCE (SHA256 checksum 70e3cdce143c45a76e155ffb5969841e0153e011f59eb9f2c6e6be0707030abf)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum 5a702fe5e50dc6055daa723df71a0874317c9ff5844eea30104587a61097ecf4)

CentOS 6

AWS CloudHSM não suporta CentOS 6 com Client SDK versão 3.4.4.

Use o the section called “Versão 3.2.1” para CentOS 6 ou escolha uma plataforma compatível.

CentOS 7 (7.8+)

Faça download do software versão 3.4.4 para o CentOS 7:

- AWS CloudHSM Cliente (SHA256 checksum 7d61d835ae38c6ce121d102b516527f342a76ac31733768097d5cab8bc482610)
- Biblioteca PKCS #11 (SHA256 checksum 2099f324ff625e1a46d96c1d5084263ca1d650424d7465ead43fe767d6687f36)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 6d8e81ad1208652904fe4b6abc4f174e866303f2302a6551c3fbef617337e663)
- Provedor JCE (SHA256 checksum 70e3cdce143c45a76e155ffb5969841e0153e011f59eb9f2c6e6be0707030abf)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum 5a702fe5e50dc6055daa723df71a0874317c9ff5844eea30104587a61097ecf4)

CentOS 8

Faça download do software versão 3.4.4 para o CentOS 8:

- AWS CloudHSM Cliente (SHA256 checksum 81639c9ec83e501709c4117ba9d98b23dea7838a206ed244c9c6cc0d65130f8c)
- Biblioteca PKCS #11 (SHA256 checksum 9a15daa87b8616cf03a6bf6b375f53451ef448dbc54bf2c27fbc2be7823fc633)
- Provedor JCE (SHA256 checksum 2b1c4208992903cf7bcc669c1392c59a64fbfc82e010c626ffa58d0cb8e9126b)

- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum 3adbecc802e0854c23aa4b8d80540d1748903c8dba93b6c8042fb7885051c360)

> **Note**
>
> Devido ao recente fim da vida útil do CentOS 8, não poderemos mais oferecer suporte a esta plataforma na próxima versão.

RHEL 6

AWS CloudHSM não é compatível com RedHat Enterprise Linux 6 com Client SDK versão 3.4.4.

Use the section called “Versão 3.2.1” para RedHat Enterprise Linux 6 ou escolha uma plataforma compatível.

RHEL 7 (7.8+)

Baixe o software versão 3.4.4 para RedHat Enterprise Linux 7:

- AWS CloudHSM Cliente (SHA256 checksum 7d61d835ae38c6ce121d102b516527f342a76ac31733768097d5cab8bc482610)
- Biblioteca PKCS #11 (SHA256 checksum 2099f324ff625e1a46d96c1d5084263ca1d650424d7465ead43fe767d6687f36)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 6d8e81ad1208652904fe4b6abc4f174e866303f2302a6551c3fbef617337e663)
- Provedor JCE (SHA256 checksum 70e3cdce143c45a76e155ffb5969841e0153e011f59eb9f2c6e6be0707030abf)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum 5a702fe5e50dc6055daa723df71a0874317c9ff5844eea30104587a61097ecf4)

RHEL 8 (8.3+)

Baixe o software versão 3.4.4 para RedHat Enterprise Linux 8:

- AWS CloudHSM Cliente (SHA256 checksum 81639c9ec83e501709c4117ba9d98b23dea7838a206ed244c9c6cc0d65130f8c)
- Biblioteca PKCS #11 (SHA256 checksum 9a15daa87b8616cf03a6bf6b375f53451ef448dbc54bf2c27fbc2be7823fc633)
- Provedor JCE (SHA256 checksum 2b1c4208992903cf7bcc669c1392c59a64fbfc82e010c626ffa58d0cb8e9126b)

- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum 3adbecc802e0854c23aa4b8d80540d1748903c8dba93b6c8042fb7885051c360)

Ubuntu 16.04 LTS

Faça download do software versão 3.4.4 para o Ubuntu 16.04 LTS:

- AWS CloudHSM Cliente (SHA256 checksum 317c92c2e0b5d60afab1beb947f053d13ddaacb994cccc2c2b898e997ece29b9)
- Biblioteca PKCS #11 (SHA256 checksum 91451c420c51488a022569fd32f052a3b988a2883ea4c2ac952acb61a2fea37c)
- Mecanismo dinâmico do OpenSSL (SHA256 checksum 4098771ad0e38df9bf14d50520ca49b9395f819f0387e2bc3b0e61abb5888e66)
- Provedor JCE (SHA256 checksum e136ff183271c2f9590a9fccb8261a7eb809506686b070e3854df1b8686c6641)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum cbf24a4032f393a913a9898b1b27036392104e8e05d911cab84049b2bcca2541)

> **Note**
>
> Devido ao EOL iminente do Ubuntu 16.04, pretendemos abandonar o suporte para esta plataforma na próxima versão.

Ubuntu 18.04 LTS

Faça download do software versão 3.4.4 para o Ubuntu 18.04 LTS:

- AWS CloudHSM Cliente (SHA256 checksum cf57d5e0e95efbf032aac8887aebd59ac8cc80e97c69e7c39fdad40873374fe8)
- Biblioteca PKCS #11 (SHA256 checksum 428f8bdad7925db5401112f707942ee8f3ca554f4ab53fa92237996e69144d2f)
- Provedor JCE (SHA256 checksum 1ff17b8f7688e84f7f0bfc96383564dca598a1cab2f2c52c888d0361682f2b9e)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum afe253046146ed6177c520b681efc680dac1048c4a95b3d8ad0f305e79bbe93e)

Windows Server

AWS CloudHSM oferece suporte às versões de 64 bits do Windows Server 2012, Windows Server 2012 R2, Windows Server 2016 e Windows Server 2019. O software cliente AWS CloudHSM 3.4.4 para Windows Server inclui os provedores de CNG e KSP necessários. Para obter detalhes, consulte Instalar e configurar o AWS CloudHSM cliente (Windows). Faça download da versão mais recente (3.4.4) do software para Windows Server:

- AWS CloudHSM para Windows Server (SHA256 checksum d51a7db588e9121d8f0b0351606bd986e1c4de6547f2c8235200dc8a5ffbe53e)
- AWS CloudHSM utilitário de gerenciamento (SHA256 checksum 0c12d7da9086735cdf189535937a8e036163009c5018dcaf2ee9cdbb6bd4c06f)

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.

Versões descontinuadas

As versões 5.8.0 e anteriores estão obsoletas. 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 obsoletas do Client SDK 5

Esta seção lista as versões obsoletas do Client SDK 5.

Versão 5.8.0

A versão 5.8.0 introduz autenticação de quórum para CLI do CloudHSM, descarregamento de SSL/TLS com JSSE, suporte a vários slots para PKCS #11, suporte a vários clusters/vários usuários para JCE, extração de chaves com JCE, KeyFactory compatível com JCE, novas configurações de repetição para códigos de retorno não terminais e inclui estabilidade aprimorada e correções de erros para todos os 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 de desempacotamento para NoPadding AES/CBC/.
- Foram adicionados novos tipos de fábricas importantes: SecretKeyFactory KeyFactory e.

CLI do CloudHSM

- Adicionado suporte para autenticação de quórum

Versão 5.7.0

A versão 5.7.0 introduz a CLI do CloudHSM 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
  - ASESCMAC

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/, para modos de criptografia e NoPadding 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
  - Modo NoPadding AES/CBC/Criptografia e Decriptografia
  - Modo de criptografia e descriptografia AES/ECB/PKCS5Padding
  - Modo NoPadding AES/CTR/Criptografia e Decriptografia
  - Modo AES/GCM/Wrap and Unwrap NoPadding
  - Modo de criptografia e descriptografia DESede/ECB/PKCS5Padding

  - Desede/CBC/Modo de Criptografia e NoPadding Decriptografia
  - Modo NoPadding AESWrap/ECB/Wrap and Unwrap
  - Modo de agrupamento e desagrupamento AESWrap/ECB/PKCS5Padding
  - Modo ZeroPadding AESWrap/ECB/Wrap and Unwrap
  - Modo de agrupamento e desagrupamento RSA/ECB/PKCS1Padding
  - Modo de agrupamento e desagrupamento RSA/ECB/OAEP
  - Modo de agrupamento e desagrupamento RSA/ECB/OAEPWithSHA-1ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSA/ECB/OAEPWithSHA-224ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSA/ECB/OAEPWithSHA-256ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSA/ECB/OAEPWithSHA-384ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSA/ECB/OAEPWithSHA-512ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSAAESWrap/ECB/OAEPPadding
  - Modo de agrupamento e desagrupamento RSAAESWrap/ECB/OAEPWithSHA-1ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSAAESWrap/ECB/OAEPWithSHA-224ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSAAESWrap/ECB/OAEPWithSHA-256ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSAAESWrap/ECB/OAEPWithSHA-384ANDMGF1Padding
  - Modo de agrupamento e desagrupamento RSAAESWrap/ECB/OAEPWithSHA-512ANDMGF1Padding
- KeyFactory e 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 (3DES)
  - GenericSecret
  - HMAC – com suporte a hash SHA1, SHA224, SHA256, SHA384, SHA512
- 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 melhora a estabilidade e inclui correções de erros para todos os 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 (3DES)
  - HMAC – com suporte a hash SHA1, SHA224, SHA256, SHA384, SHA512.
- Cifras (somente criptografar e descriptografar)
  - AES/GCM/ NoPadding
  - AES/BCE/ NoPadding

- AES/CBC/PKCS5Padding
- Desede/ECB/ NoPadding
- DESede/CBC/PKCS5Padding
- AES/CTR/ NoPadding
- RSA/ECB/PKCS1Padding
- RSA/ECB/OAEPPadding
- RSA/ECB/OAEPWithSHA-1ANDMGF1Padding
- RSA/ECB/OAEPWithSHA-224ANDMGF1Padding
- RSA/ECB/OAEPWithSHA-256ANDMGF1Padding
- RSA/ECB/OAEPWithSHA-384ANDMGF1Padding
- RSA/ECB/OAEPWithSHA-512ANDMGF1Padding

- 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 5.3.0

Biblioteca PKCS #11

- Maior estabilidade e correção de erros.

Mecanismo dinâmico do OpenSSL

- Adicione suporte para assinatura/verificação ECDSA 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 (3DES)
  - HMAC – com suporte a hash SHA1, SHA224, SHA256, SHA384, SHA512.
- Cifras (somente criptografar e descriptografar)
  - AES/GCM/ NoPadding
  - AES/BCE/ NoPadding
  - AES/CBC/PKCS5Padding
  - Desede/ECB/ NoPadding
  - DESede/CBC/PKCS5Padding
  - AES/CTR/ NoPadding
  - RSA/ECB/PKCS1Padding

  - RSA/ECB/OAEPPadding
  - RSA/ECB/OAEPWithSHA-1ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-224ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-256ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-384ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-512ANDMGF1Padding
- 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

- 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
- CKM_DES3_KEY_GEN
- CKM_DES3_CBC
- CKM_DES3_CBC_PAD
- CKM_DES3_ECB
- 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
- 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
- CKM_AES_ECB
- CKM_AES_CTR
- CKM_AES_CBC
- CKM_AES_CBC_PAD
- CKM_SP800_108_COUNTER_PDF
- CKM_GENERIC_SECRET_KEY_GEN
- CKM_SHA_1_HMAC
- CKM_SHA224_HMAC
- CKM_SHA256_HMAC
- CKM_SHA384_HMAC
- CKM_SHA512_HMAC
- CKM_RSA_PKCS_OAEP somente para agrupamento/desagrupamento
- CKM_RSA_AES_KEY_WRAP
- CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD
- CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD
- 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 APIs do OpenSSL:
  - Geração de chave de RSA para chaves de 2048, 3072 e 4096 bits.
  - APIs do OpenSSL:
    - Assinatura RSA 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.
- Suporte para descarregamento SSL/TLS no CentOS 8.3+, Red Hat Enterprise Linux (RHEL) 8.3 e Ubuntu 18.04 LTS, incluindo NGINX 1.19 (para conjuntos de criptografia selecionados).

  Para obter mais informações, consulte Descarregamento de SSL/TLS no Linux.

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
- CKM_SHA1_RSA_PKCS
- CKM_SHA224
- CKM_SHA224_RSA_PKCS
- CKM_SHA256
- CKM_SHA256_RSA_PKCS
- CKM_SHA384
- CKM_SHA384_RSA_PKCS
- CKM_SHA512

- CKM_SHA512_RSA_PKCS

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:

- Generate keys (AES, RSA, EC)
- List key attributes

- Criptografar e descriptografar dados com AES-GCM
- Sign and verify data with RSA

Versões obsoletas do Client SDK 3

Esta seção lista as versões obsoletas do Client SDK 3.

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 ter mais informações, consulte ???.

Mecanismo dinâmico do OpenSSL

- Não suporta para CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Para ter mais informações, consulte ???.

Provedor JCE

- Adicione suporte de plataforma para CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Para ter 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.
  - Desabilitada a capacidade de especificar um IV personalizado ao empacotar uma chave usando AES_KEY_WRAP_PAD_PKCS5. 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 DE QUEBRA: os pacotes de criptografia TLS 1.0 e 1.2 com SHA1 não estão disponíveis no mecanismo OpenSSL 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

- ALTERAÇÃO DE QUEBRA: para resolver um problema com a conformidade com Java Cryptography Extension (JCE), o empacotamento e desempacotamento AES agora usa corretamente o algoritmo AESWrap em vez do algoritmo AES. Isto significa que `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE` não é mais bem-sucedido para os mecanismos AES/BCE e AES/CBC.

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 encapsulou chaves com a versão de cliente 3.0.0 ou anterior, na versão 3.1.1, será necessário usar AESWrap/ECB/PKCS5Padding para desencapsular 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.

nd-of-life Lançamentos E

AWS CloudHSM anuncia o fim da vida útil de versões que não são mais compatíveis com o serviço. Para preservar a segurança de seu aplicativo, nos reservamos o direito de recusar ativamente conexões de end-of-life lançamentos.

- Atualmente, nenhuma versão do SDK do cliente é end-of-life lançada.

Histórico do documento

Este tópico descreve atualizações importantes no Guia do usuário do AWS CloudHSM .

Tópicos

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
Novo tipo de HSM e modo de cluster	Lançou 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. AWS 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 capacidad e 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 capacidad e 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 Enterpris e 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	Adição de autenticação de quorum (controle de acesso M de N) para responsáveis pela criptografia (COs). Para ter mais informações, consulte Como usar o CloudHSM Management Utility (CMU) para gerenciar a autentica ção de quórum (controle de acesso M ou N).	9 de novembro de 2017
Atualizar	Adição de uma documentação sobre como usar a ferrament a de linha de comando key_mgmt_util. Para ter mais informações, consulte	9 de novembro de 2017

Alteração	Descrição	Data
	referência do comando key_mgmt_util.	
Novo conteúdo	Adição do Oracle Transparent Data Encryption. Para ter mais informações, consulte Criptografia do Oracle Database.	25 de outubro de 2017
Novo conteúdo	Adição de descarregamento SSL. Para ter 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.