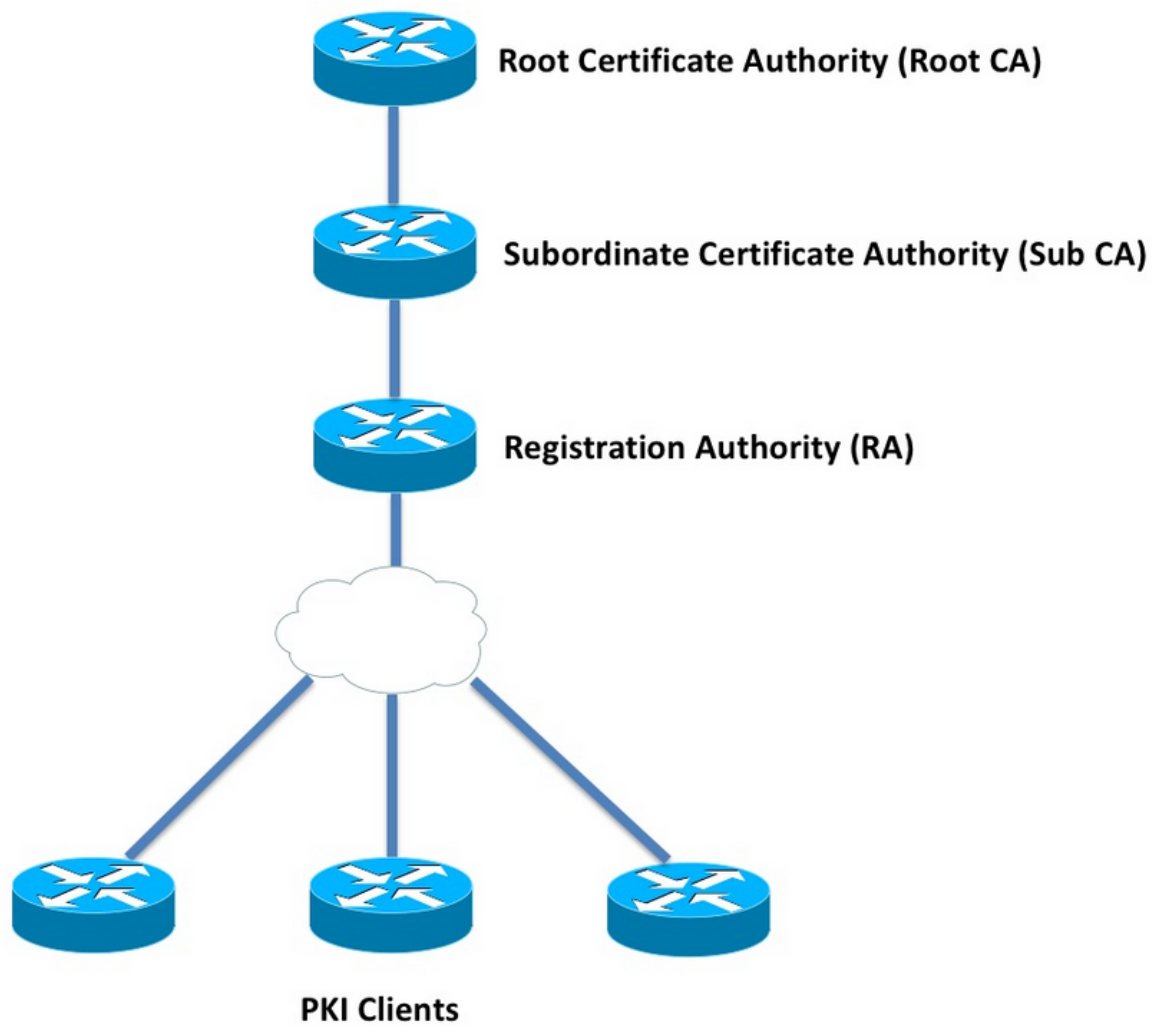
Guia de implantação de PKI do IOS: Projeto e implantação iniciais

Contents

Introduction

Este documento descreve as funcionalidades do servidor PKI IOS e do cliente em detalhes. Ele aborda o projeto inicial do PKI do IOS e as considerações de implantação.

Infraestrutura PKI



Autoridade de certificação

A autoridade de certificação (AC), também conhecida como servidor PKI em todo o documento, é

uma entidade confiável que emite certificados. O PKI é baseado em confiança e a hierarquia de confiança é iniciada na Autoridade de Certificação Raiz (AC raiz). Como o CA raiz está no topo da hierarquia, ele tem um certificado autoassinado.

Autoridade de certificação subordinada

Na hierarquia de confiança PKI, todas as autoridades de certificado abaixo de Raiz são conhecidas como Autoridades de Certificação Subordinada (Sub-CA). Evidentemente, um certificado Sub-CA é emitido pela CA, que está um nível acima.

O PKI não impõe nenhum limite ao número de Sub-CAs em uma determinada hierarquia. No entanto, em uma implantação empresarial com mais de 3 níveis de autoridades de certificado pode se tornar difícil de gerenciar.

Autoridade de registro

O PKI define uma autoridade de certificação especial conhecida como Autoridade de Registro (RA), que é responsável por autorizar os clientes PKI a se inscreverem em uma determinada Sub-CA ou Root-CA. O RA não emite certificados para clientes PKI, em vez disso, decide qual PKI-Client pode ou não receber um certificado pela Sub-CA ou pela AC raiz.

A função principal de um RA é descarregar a validação da solicitação de certificado de cliente básico da CA e proteger a CA da exposição direta aos clientes. Dessa forma, o RA fica entre os clientes PKI e a CA, protegendo assim a CA de qualquer tipo de ataque de negação de serviço.

Cliente PKI

Qualquer dispositivo que solicite um certificado baseado em um par de chaves público-privado residente para provar sua identidade para outros dispositivos é conhecido como cliente PKI.

Um cliente PKI deve ser capaz de gerar ou armazenar um par de chaves público-privadas, como RSA ou DSA ou ECDSA.

Um certificado é uma prova de identidade e validade de uma determinada chave pública, desde que a chave privada correspondente exista no dispositivo.

Servidor PKI IOS

Tabela 1. Evolução do recurso do servidor PKI IOS

Recurso	IOS [ISR-G1, ISR-G2]	IOS-XE [ASR1K, ISR4K]
Servidor IOS CA/PKI	12.3(4)T	XE 3.14.0 / 15.5(1)S
Rollover de Certificado do Servidor PKI IOS	12.4(1)T	XE 3.14.0 / 15.5(1)S
IOS PKI HA	15,0 (1) M	NA [Redundância Inter-RP implícita disponível]
IOS RA para CA de terceiros	15.1(3)T	XE 3.14.0 / 15.5(1)S

Antes de entrar na configuração do servidor PKI, o administrador deve entender esses conceitos principais.

Fonte de tempo autoritativa

Um dos fundamentos da infraestrutura de PKI é o Time. O relógio do sistema define se um certificado é válido ou não. Portanto, no IOS, o relógio deve ser tornado autoritativo ou confiável. Sem uma fonte de tempo autoritativa, o servidor PKI pode não funcionar como esperado, e é altamente recomendável tornar o relógio no IOS autoritativo usando estes métodos:

NTP (Network Time Protocol)

Sincronizar o relógio do sistema com um Servidor de Tempo é a única maneira verdadeira de tornar o relógio do sistema confiável. Um roteador IOS pode ser configurado como um cliente NTP para um servidor NTP conhecido e estável na rede:

```
configure terminal
ntp server <NTP Server IP address>
ntp source <source interface name>
ntp update-calendar

!! optional, if the NTP Server requires the clients to authenticate themselves
ntp authenticate
ntp authentication-key 1 md5 <key>

!! optionally an access-list can be configured to restrict time-updates from a specifc NTP
server
access-list 1 permit <NTP Server IP address>
ntp access-group peer 1
```

O IOS também pode ser configurado como um servidor NTP, que marcará o relógio do sistema local como autoritativo. Na implantação PKI em pequena escala, o servidor PKI pode ser configurado como um servidor NTP para seus clientes PKI:

```
configure terminal
ntp master <stratum-number>

!! optionally, NTP authentication can be enforced
ntp authenticate
ntp authentication-key 1 md5 <key-1>
ntp authentication-key 2 md5 <key-2>
ntp authentication-key 2 md5 <key-2>
ntp trusted-key 1 - 3

!! optinally, an access-list can be configured to restrict NTP clients
!! first allow the local router to synchronize with the local time-server
access-list 1 permit 127.127.7.1
ntp access-group peer 1

!! define an access-list to which the local time-server will serve time-synchronization services
access-list 2 permit <NTP-Client-IP>
ntp access-group serve-only 2
```

Marcação do relógio de hardware como confiável

No IOS, o relógio do hardware pode ser marcado como autoritativo usando:

```
config terminal
clock calendar-valid
```

Isso pode ser configurado junto com o NTP, e o principal motivo para fazer isso é manter a autoridade do relógio do sistema quando um roteador é recarregado, por exemplo, devido a uma falta de energia, e os servidores NTP não são acessíveis. Neste estágio, os temporizadores PKI pararão de funcionar, o que por sua vez leva a falhas de renovação/rollover de certificado. **o clock calendário-valid** atua como uma salvaguarda em tais situações.

Ao configurar isso, é importante entender que o relógio do sistema ficará dessincronizado se a bateria do sistema falhar, e o PKI começará a confiar em um relógio dessincronizado. No entanto, é relativamente mais seguro configurá-lo, do que não ter uma fonte de tempo autoritativa.

> **Note**: comando **clock day-valid** foi adicionado no IOS-XE versão XE 3.10.0 / 15.3(3)S adiante.

Nome de host e nome de domínio

Recomenda-se configurar um nome de host e um nome de domínio no Cisco IOS como uma das primeiras etapas antes de configurar qualquer serviço relacionado a PKI. O nome de host e o nome de domínio do roteador são usados nos seguintes cenários:

- O nome de par de chaves RSA padrão é derivado combinando o nome do host e o nome de domínio
- Ao se inscrever em um certificado, o nome de assunto padrão consiste em um atributo de nome de host e um nome não estruturado, que é nome de host e nome de domínio juntos.

Quanto ao servidor PKI, o nome de host e o nome de domínio não são usados:

- O nome do par de chaves padrão será o mesmo do nome do servidor PKI
- O nome de assunto padrão consiste em CN, que é o mesmo do nome do servidor PKI.

A recomendação geral é configurar um nome de host apropriado e um nome de domínio.

```
config terminal
hostname <string>
ip domain name <domain>
```

Servidor HTTP

O Servidor PKI do IOS só é ativado se o Servidor HTTP estiver ativado. É importante observar que, se o servidor PKI estiver desabilitado devido ao servidor HTTP estar desabilitado, ele poderá continuar a conceder solicitações offline [via terminal]. O recurso Servidor HTTP é necessário para processar solicitações SCEP e enviar respostas SCEP.

O IOS HTTP Server está ativado usando:

```
ip http server
```

E a porta padrão do servidor HTTP pode ser alterada de 80 para qualquer número de porta válido usando:

```
ip http port 8080
```

Conexão máxima HTTP

Um dos gargalos, enquanto implanta o IOS como servidor PKI usando o SCEP, são o Máximo de conexões HTTP simultâneas e a média de conexões HTTP por minuto.
Atualmente, o máximo de conexões simultâneas em um IOS HTTP Server é limitado a 5 por padrão e pode ser aumentado para 16, o que é altamente recomendado em uma implantação de escala média:

```
ip http max-connections 16
```

Essas instalações do IOS permitem conexões HTTP simultâneas máximas de até 1000:

- UniversalK9 IOS com uck9 conjunto de licenças

A CLI é alterada automaticamente para aceitar um argumento numérico entre 1 e 1000

```
ip http max-connections 1000
```

O servidor IOS HTTP permite 80 conexões por minuto [580 no caso de versões do IOS em que o máximo de sessões simultâneas HTTP pode ser aumentado para 1000] e quando esse limite é atingido em um minuto, o ouvinte IOS HTTP começa a limitar as conexões HTTP de entrada ao desligar o ouvinte por 15 segundos. Isso faz com que as solicitações de conexão do cliente sejam descartadas devido ao **limite de fila de conexão** TCP **atingido**. Mais informações sobre este assunto podem ser encontradas aqui

RSA Key-pair

O par de chaves RSA para a funcionalidade do servidor PKI no IOS pode ser gerado automaticamente ou gerado manualmente.
Ao configurar um servidor PKI, o IOS cria automaticamente um ponto de confiança com o mesmo nome do servidor PKI para armazenar o certificado do servidor PKI.

Gerando manualmente um par de chaves RSA do servidor PKI:

Etapa 1. Crie um par de chaves RSA com o mesmo nome do servidor PKI:

```
crypto key generate rsa general-keys label <LABEL> modulus 2048
```

Etapa 2. Antes de habilitar o servidor PKI, modifique o ponto de confiança do servidor PKI:

```
crypto pki trustpoint <PKI-SERVER-Name>
  rsakeypair <LABEL>
```

> **Note**: O valor do módulo de par de chaves RSA mencionado no ponto de confiança do servidor PKI não é considerado até o IOS ver 15.4(3)M4, e esse é um alerta conhecido. O módulo de chave padrão é 1024 bits.

Par-chave RSA de servidor PKI de geração automática:

Ao habilitar o servidor PKI, o IOS gera automaticamente um par de chaves RSA com o mesmo nome do servidor PKI, e o tamanho do módulo de chave é 1024 bits.

A partir do IOS versão 15.4(3)M5, essa configuração cria um par de chaves RSA com **<LABEL>** como o nome e a força da chave serão conforme o módulo **<MOD>** definido.

```
crypto pki trustpoint <PKI-SERVER-Name>
rsakeypair <LABEL> <MOD>
```

Spoiler

O servidor PKI CSCuu73408 do IOS deve permitir o tamanho de chave não padrão para o certificado de transferência.

O servidor PKI do IOS CSCuu73408 deve permitir o tamanho de chave não padrão para o certificado de transferência.

O padrão atual do setor é usar um mínimo de 2048 bits de par de chaves RSA.

Consideração do temporizador de rollover automático

Atualmente, o IOS PKI Server não gera um certificado de rollover por padrão e ele deve ser explicitamente ativado no servidor PKI usando o comando **auto-rollover <days-before-expire>**. Mais informações sobre a transferência do certificado são explicadas em

Esse comando especifica quantos dias antes da expiração do certificado do Servidor PKI/CA caso o IOS crie um certificado CA rollover. Observe que o certificado CA de transferência é ativado quando o certificado CA ativo atual expira. O valor padrão atualmente é 30 dias. Esse valor deve ser definido para um valor razoável dependendo do tempo de vida do certificado CA, o que, por sua vez, influencia a configuração do temporizador de inscrição automática no cliente PKI.

> **Note**: O temporizador de rollover automático deve sempre disparar antes do temporizador de inscrição automática no cliente durante a rollover de certificado de cliente e CA [conhecido como]

considerações de CRL

A infraestrutura de PKI do IOS suporta duas formas de distribuição de CRL:

Publicar a CRL em um servidor HTTP

O Servidor PKI do IOS pode ser configurado para publicar o arquivo CRL em um local específico em um Servidor HTTP usando este comando no Servidor PKI:

```
crypto pki server <PKI-SERVER-Name>
 database crl publish <URL>
```

E o servidor PKI pode ser configurado para incorporar este local CRL em todos os certificados de cliente PKI usando este comando no Servidor PKI:

```
crypto pki server <PKI-SERVER-Name>
cdp-url <CRL file location>
```

Método GetCRL do SCEP

O IOS PKI Server armazena automaticamente o arquivo CRL no local de banco de dados específico, que por padrão é a nvram, e é altamente recomendável manter uma cópia em um servidor SCP/FTP/TFTP usando este comando no PKI Server:

```
crypto pki server <PKI-SERVER-Name>
database url <URL>
or
database crl <URL>
```

Por padrão, o IOS PKI Server não incorpora o local CDP nos certificados de cliente PKI. Se os clientes PKI do IOS estiverem configurados para executar a verificação de revogação, mas o certificado que está sendo validado não tiver um CDP incorporado nele, e o ponto confiável da CA de validação estiver configurado com o local da CA (usando http://<CA-Server-IP ou FQDN>), o IOS retorna ao método GetCRL baseado em SCEP por padrão.
O SCEP GetCRL executa a recuperação de CRL executando HTTP GET neste URL:

```
http:://<CA-Server-IP/FQDN>/cgi-bin/pkiclient.exe?operation=GetCRL
```

> **Note**: Na CLI do IOS, antes de entrar **?**, pressione **Ctrl + V** sequência de teclas.

O IOS PKI Server também pode incorporar este URL como o local do CDP. A vantagem de fazer isso é duas vezes:

- Garante que todos os clientes PKI não baseados em SCEP do IOS possam executar a

recuperação de CRL.

- Sem um CDP incorporado, as mensagens de solicitação GetCRL do IOS SCEP são assinadas (usando um certificado autoassinado temporário) conforme definido no rascunho do SCEP. No entanto, as solicitações de recuperação de CRL não precisam ser assinadas e, incorporando o URL do CDP para o método GetCRL, a assinatura das solicitações de CRL pode ser evitada.

Vida útil do CRL

A vida útil da CRL do IOS PKI Server pode ser controlada usando este comando no Servidor PKI:

```
crypto pki server <PKI-SERVER-Name>
 lifetime crl <0 - 360>
```

O valor é em horas. Por padrão, o tempo de vida da CRL é definido como 6 horas. Dependendo da frequência com que os certificados são revogados, o ajuste do tempo de vida da CRL para um valor ótimo aumenta o desempenho de recuperação da CRL na rede.

Considerações do banco de dados

O IOS PKI Server usa a nvram como local de banco de dados padrão e é altamente recomendável usar um servidor FTP ou TFTP ou SCP como local de banco de dados. Por padrão, o IOS PKI Server cria dois arquivos:

- <Server-Name>.ser - Contém o último número de série emitido pela CA em hexadecimal. O arquivo está em formato de texto simples e contém estas informações:
  db_version = 1
  last_serial = 0x4

- <Server-Name>.crl - Este é o arquivo CRL codificado por DER publicado pela CA

O IOS PKI Server armazena informações no banco de dados em 3 níveis configuráveis:

- Mínimo - Este é o nível padrão e, nesse nível, nenhum arquivo é criado no banco de dados e, portanto, nenhuma informação está disponível no servidor CA sobre os certificados de cliente concedidos no passado.
- Nomes - Neste nível, o servidor PKI do IOS cria um arquivo chamado <Serial-Number>.cnm para cada certificado de cliente emitido, em que o nome <Serial-Number> refere-se ao número de série do certificado de cliente emitido E este arquivo cnm contém o nome do assunto e a data de expiração do certificado de cliente.
- Concluído - Nesse nível, o IOS PKI Server cria dois arquivos para cada certificado de cliente emitido:
- <Número de série>.cnm
- <Número de série>.crt

aqui, o arquivo crt é o arquivo de certificado do cliente, que é codificado por DER.

Esses pontos são importantes:

- Antes de emitir um certificado de cliente, o IOS PKI Server refere-se a <Server-Name>.ser para determinar e derivar o número de série do certificado.
- Com o nível do banco de dados definido como Nomes ou Completo, <Número de série>.cnm e <Número de série>.crt precisam ser gravados no banco de dados antes de enviar o certificado concedido/emitido ao cliente
- Com o url do banco de dados definido como Nomes ou Concluído, o URL do banco de dados deve ter espaço suficiente para salvar os arquivos. Portanto, a recomendação é configurar um servidor de arquivos externo [FTP, TFTP ou SCP] como o URL do banco de dados.
- Com a URL do banco de dados externo configurada, é absolutamente necessário garantir que o servidor de arquivos esteja acessível durante o processo de concessão de certificado, o que, caso contrário, marcaria o servidor de CA como desabilitado. E a intervenção manual é necessária para colocar o servidor CA novamente on-line.

Arquivo de banco de dados

Ao implantar um servidor PKI, é importante considerar os cenários de falha e estar preparado, caso haja uma falha de hardware. Há duas maneiras de conseguir isso:

1. Redundância
   Nesse caso, dois dispositivos ou unidades de processamento atuam como Ative-Standby para fornecer redundância.
   A alta disponibilidade do IOS PKI Server pode ser alcançada usando dois roteadores ISR habilitados para HSRP [ISR G1 e ISR G2], conforme explicado na

   Os sistemas baseados em IOS XE [ISR4K e ASR1k] não têm opção de redundância de dispositivo disponível. No entanto, no ASR1k, a redundância entre RP está disponível por padrão.
2. Arquivando o par de chaves e arquivos do servidor CA
   O IOS fornece um recurso para arquivar o par de chaves do servidor PKI e o certificado. O arquivamento pode ser feito com dois tipos de arquivos:
   PEM - O IOS cria arquivos formatados PEM para armazenar a chave pública RSA, a chave privada RSA criptografada, o certificado do servidor CA. Rollover Key-pair e os certificados são arquivados automaticamentePKCS12 - O IOS cria um único arquivo PKCS12 contendo o certificado do servidor CA e a chave privada RSA correspondente criptografada usando uma senha.

O arquivamento do banco de dados pode ser ativado usando este comando no Servidor PKI:

```
crypto pki server <PKI-SERVER-Name>
 database archive {pkcs12 | pem} password <password>
```

Também é possível armazenar os arquivos arquivados em um servidor separado, possivelmente usando um protocolo seguro (SCP) usando o seguinte comando no Servidor PKI:

```
crypto pki server <PKI-SERVER-Name>
 database url {p12 | pem} <URL>
```

De todos os arquivos no banco de dados, exceto os arquivos arquivados e o arquivo .Ser, todos os outros arquivos estão em texto claro e não representam nenhuma ameaça real se forem

perdidos e, portanto, podem ser armazenados em um servidor separado sem incorrer em muita sobrecarga ao gravar os arquivos, por exemplo, um servidor TFTP.

IOS como Sub-CA

Por padrão, o IOS PKI Server assume a função de CA raiz. Para configurar um servidor PKI subordinado (Sub-CA), primeiro ative este comando na seção de configuração do servidor PKI (antes de ativar o servidor PKI):

```
crypto pki server <Sub-PKI-SERVER-Name>
 mode sub-cs
```

Usando isso, configure a URL do Root-CA no ponto de confiança do PKI Server:

```
crypto pki trustpoint <Sub-PKI-SERVER-Name>
 enrollment url <Root-CA URL>
```

Ativar este servidor PKI agora aciona estes eventos:

- O ponto confiável do servidor PKI é autenticado para instalar o certificado de CA raiz.
- Depois que a CA raiz é autenticada, o IOS gera um CSR para a restrição básica Subordinate-CA [x509 contendo o sinalizador CA:TRUE] e o envia para a CA raiz

Independentemente do modo de concessão configurado na CA raiz, o IOS coloca as solicitações de certificado CA (ou RA) na fila pendente. Um administrador precisa conceder manualmente os certificados CA.
Para exibir a solicitação de certificado pendente e a ID de solicitação:

```
show crypto pki server <Server-Name> requests
```

Para conceder a solicitação:

```
crypto pki server <Server-Name> grant <request-id>
```

- Usando isso, a operação subsequente SCEP POLL (GetCertInitial) faz o download do certificado Sub-CA e o instala no roteador, o que ativa o Servidor PKI Subordinado

IOS como RA

O Servidor PKI de E/S pode ser configurado como uma Autoridade de Registro para uma AC de Sub-Coordenada ou Raiz. Para configurar o Servidor PKI como uma autoridade de registro, primeiro habilite este comando na seção de configuração do Servidor PKI (antes de habilitar o Servidor PKI):

```
crypto pki server <RA-SERVER-Name>
 mode ra
```

Depois disso, configure a URL da CA no ponto de confiança do servidor PKI. Indica qual CA está protegida pelo RA:

```
crypto pki trustpoint <RA-SERVER-Name>
```

```
 enrollment url <CA URL>
 subject-name CN=<Common Name>, OU=ioscs RA, OU=TAC, O=Cisco
```

Uma autoridade de registro não emite certificados, portanto a configuração **nome-emitente** sob o RA não é necessária e não é eficaz mesmo que esteja configurada. O nome do assunto de um RA é configurado no ponto confiável do RA usando o comando **subject-name**. É importante configurar **OU = ioscs RA** como parte do nome do assunto para que a CA do IOS identifique o IOS RA, ou seja, identifique as solicitações de certificado autorizadas pelo IOS RA.

O IOS pode atuar como uma Autoridade de Registro para CAs de terceiros, como o Microsoft CA, e para manter a compatibilidade, o IOS RA deve ser ativado usando este comando na seção de configuração do PKI Server (antes de ativar o PKI Server):

```
mode ra transparent
```

No modo RA padrão, o IOS assina as solicitações do cliente [PKCS#10] usando o certificado RA. Esta operação indica ao IOS PKI Server que a solicitação de certificado foi autorizada por um RA.

Com o modo RA transparente, o IOS encaminha as solicitações do cliente em seu formato original sem introduzir o certificado RA, e isso é compatível com o Microsoft CA como um exemplo bem conhecido.

Cliente PKI do IOS

Uma das entidades de configuração mais importantes no cliente PKI do IOS é um ponto de confiança. Os parâmetros de configuração do ponto de confiança são explicados em detalhes nesta seção.

Fonte de tempo autoritativa

Como observado anteriormente, a fonte de tempo autoritativa também é um requisito para o cliente PKI. O cliente PKI do IOS pode ser configurado como um cliente NTP usando estas configurações:

```
configure terminal
ntp server <NTP Server IP address>
ntp source <source interface name>
ntp update-calendar

!! optional, if the NTP Server requires the clients to authenticate themselves
ntp authenticate
ntp authentication-key 1 md5 <key>

!! Optionally an access-list can be configured to restrict time-updates from a specifc NTP
server
access-list 1 permit <NTP Server IP address>
ntp access-group peer 1
```

Nome de host e nome de domínio

Uma recomendação geral é configurar um nome de host e um nome de domínio no Roteador:

```
configure terminal
hostname <string>
```

```
ip domain name <domain>
```

Par-chave RSA

No cliente PKI do IOS, o par de chaves RSA para uma determinada inscrição de ponto de confiança pode ser gerado automaticamente ou gerado manualmente.

O processo automático de geração de chaves RSA envolve o seguinte:

- O IOS por padrão cria um par de chaves RSA de 512 bits
- O nome de par de chaves gerado automaticamente é hostname.domain-name, que é o nome de host do dispositivo combinado com o nome de domínio do dispositivo
- O par de chaves gerado automaticamente não está marcado como exportável.

O processo automático de geração de chaves RSA envolve o seguinte:

- Opcionalmente, um par de chaves RSA de finalidade geral de uma força adequada pode ser gerado manualmente usando:
- ```
  crypto key generate rsa general-keys label <LABEL> modulus < MOD> [exportable]
  ```
  Aqui, LABEL - o nome do par de chaves RSA
  MOD - Módulo de chave RSA ou força em bits entre 360 e 4096, que é tradicionalmente 512, 1024, 2048 ou 4096.
  A vantagem de gerar manualmente o par de chaves RSA é a capacidade de marcar o par de chaves como exportável, o que, por sua vez, permite que o certificado de identidade seja completamente exportado, que pode ser restaurado em outro dispositivo. No entanto, é preciso entender as implicações de segurança dessa ação.
- Um par de chaves RSA é vinculado a um ponto de confiança antes da inscrição usando este comando

  ```
  crypto pki trustpoint MGMT
  rsakeypair <LABEL> [<MOD> <MOD>]
  ```
  Aqui, se um par de chaves RSA chamado **<LABEL>** já existir, ele será atendido durante a inscrição no ponto de confiança.
  Se um par de chaves RSA chamado **<LABEL>** não existir, uma das seguintes ações será executada durante a inscrição:
  - Se nenhum argumento **<MOD>** for passado, um par de chaves de 512 bits chamado **<LABEL>** será gerado.
  - se um argumento **<MOD>** for passado, um par de chaves **<MOD>** de propósito geral de bits chamado **<LABEL>** será gerado
  - se dois argumentos **<MOD>** forem passados, um par de chaves de assinatura de bits **<MOD>** e um par de chaves de criptografia de bits **<MOD>** serão gerados

Ponto de confiança

Um ponto de confiança é um contêiner abstrato para conter um certificado no IOS. Um único ponto confiável é capaz de armazenar dois certificados ativos em um determinado momento:

- Um certificado CA - Carregar um certificado CA em um determinado ponto confiável é
  ```

conhecido como processo de autenticação de ponto confiável.
- Um certificado de ID emitido pela CA - Carregando ou Importando um certificado de ID para um determinado ponto de confiança é conhecido como processo de inscrição de ponto de confiança.

Uma configuração de ponto de confiança é conhecida como uma política de confiança, e isso define que:

- Qual certificado CA está carregado no ponto de confiança?
- Para qual CA o ponto confiável se inscreve?
- Como o IOS registra o ponto de confiança?
- Como um certificado emitido pela AC [carregada no ponto de confiança] é validado?

Os principais componentes de um ponto de confiança são explicados aqui.

Modo de inscrição

Um modo de registro de ponto de confiança, que também define o modo de autenticação de ponto de confiança, pode ser executado através de 3 métodos principais:

1. Inscrição de terminal - método manual de executar autenticação de ponto de confiança e inscrição de certificado usando o copy-paste no terminal CLI.
2. Inscrição no SCEP - autenticação e inscrição de ponto de confiança usando SCEP sobre HTTP.
3. Perfil de inscrição - Aqui, os métodos de autenticação e inscrição são definidos separadamente. Juntamente com os métodos de registro de terminal e SCEP, os perfis de inscrição fornecem uma opção para especificar comandos HTTP/TFTP para executar a recuperação de arquivos a partir do Servidor, que é definido usando uma URL de autenticação ou inscrição no perfil.

Interface de origem e VRF

A autenticação e a inscrição de ponto de confiança sobre HTTP (SCEP) ou TFTP (Enrollment Profile) usa o sistema de arquivos do IOS para executar operações de e/s de arquivos. Essas trocas de pacotes podem ser originadas de uma interface de origem específica e de um VRF.

No caso de configuração de ponto confiável clássico, essa funcionalidade é habilitada usando **interface de origem** e subcomandos **vrf** no ponto confiável.

No caso de perfis de inscrição, **interface de origem** e **inscrição | os** comandos **authentication url <http/tftp://Server-location> vrf <vrf-name>** fornecem a mesma funcionalidade.

Exemplo de configuração:

```
vrf definition MGMT
 rd 1:1
 address-family ipv4
 exit-address-family

crypto pki trustpoint MGMT
 source interface Ethernet0/0
 vrf MGMT
```

or

```
crypto pki profile enrollment MGMT-Prof
 enrollment url http://10.1.1.1:80 vrf MGMT
 source-interface Ethernet0/0
crypto pki trustpoint MGMT
 enrollment profile MGMT-Prof
```

Inscrição e renovação automáticas de certificados

O cliente PKI do IOS pode ser configurado para executar a inscrição e renovação automáticas usando este comando na seção de ponto de confiança PKI:

```
crypto pki trustpoint MGMT
auto-enroll <percentage> <regenerate>
```

Aqui, o **comando autoenroll <percentual> [regenerar]** afirma que o IOS deve executar a renovação de certificado exatamente em 80% do tempo de vida do certificado atual.

A palavra-chave **regenerate** afirma que o IOS deve regenerar o par de chaves RSA conhecido como par de chaves de sombra durante cada operação de renovação de certificado.

Este é o comportamento de inscrição automática:

- No momento em que a **inscrição automática** é configurada, se o ponto de confiança for autenticado, o IOS executará uma inscrição automática no servidor localizado na URL mencionada como parte do comando **enrollment url** na seção ponto de confiança PKI ou no perfil de inscrição.
- No momento em que um ponto de confiança é inscrito com um Servidor PKI ou uma AC, um temporizador RENEW ou SHADOW é inicializado no cliente PKI com base na porcentagem **de inscrição automática** do certificado de identidade atual instalado sob o ponto de confiança. Esse temporizador é visível no comando **show crypto pki timer**. Mais informações sobre as funções do temporizador *referem-se a*
- O suporte à capacidade de renovação vem do servidor PKI. Mais sobre isso em
  O IOS PKI Client executa dois tipos de renovação:
  Renovação implícita: Se o servidor PKI não enviar "Renovação" como um recurso suportado, o IOS executa uma inscrição inicial na porcentagem de inscrição automática definida. Ou seja, o IOS usa um certificado autoassinado para assinar a solicitação de renovação.Renovação explícita: Quando o PKI Server suporta o recurso de renovação de certificado de cliente PKI, ele anuncia "Renovação" como um recurso suportado. O IOS leva esse recurso em consideração durante a renovação do certificado, ou seja, o IOS usa o certificado de identidade ativo atual para assinar a solicitação do certificado de renovação.

Deve-se tomar cuidado ao configurar o percentual de inscrição automática. Em qualquer cliente PKI na implantação, se surgir uma condição em que o certificado de identidade expire ao mesmo tempo que o certificado CA emissor, o valor da inscrição automática sempre acionará a operação de renovação [sombra] depois que a CA tiver criado o certificado de transferência. *Consulte a* seção **dependências do Temporizador PKI** em

Verificação de revogação de certificado

Um ponto confiável de PKI autenticado, ou seja, um ponto confiável de PKI que contém um certificado de CA, é capaz de executar a validação do certificado durante uma negociação de IKE ou SSL, em que o certificado de peer é submetido a uma validação de certificado completa. Um dos métodos de validação é verificar o status de revogação de certificado de peer usando um dos dois métodos a seguir:

- Lista de revogação de certificado (CRL) - Este é um arquivo que contém os números de série dos certificados revogados por uma determinada CA. Este arquivo é assinado usando o certificado CA emissor. O método CRL envolve o download do arquivo CRL usando HTTP ou LDAP.

- Protocolo de Status de Certificado Online (OCSP - Online Certificate Status Protocol) - O IOS estabelece um canal de comunicação com uma entidade chamada OCSP Responder, que é um servidor designado pela CA emissora. Um cliente como o IOS envia uma solicitação contendo o número de série do certificado que está sendo validado. O respondente do OCSP responde com o status de revogação do número de série especificado. O canal de comunicação pode ser estabelecido usando qualquer protocolo de aplicação/transporte suportado, que geralmente é HTTP.

A verificação de revogação pode ser definida usando estes comandos na seção de ponto confiável PKI:

```
crypto pki trustpoint MGMT
 revocation-check crl ocsp none
```

Por padrão, um ponto confiável é configurado para executar verificação de revogação usando crl.

Os métodos podem ser reordenados e a verificação de status da revogação é executada na ordem definida. O método "none" ignora a verificação de revogação.

cache de CRL

Com a verificação de revogação baseada em CRL, cada validação de certificado pode disparar um novo download de arquivo de CRL. E à medida que o arquivo CRL aumenta ou se o ponto de distribuição da CRL (CDP) está mais distante, fazer o download do arquivo durante cada processo de validação dificulta o desempenho do protocolo dependente da validação do certificado. Portanto, o cache de CRL é executado para melhorar o desempenho, e o cache da CRL leva em consideração a validade da CRL.

A validade da CRL é definida usando dois parâmetros de tempo: **LastUpdate**, que é a última vez que a CRL foi publicada pela AC emissora, e **NextUpdate**, que é o momento em que uma nova versão do arquivo CRL é publicada pela AC emissora.

O IOS armazena em cache todas as CRL baixadas enquanto a CRL for válida. No entanto, em certas circunstâncias, como o CDP não ser alcançado temporariamente, pode ser necessário manter o CRL no cache por um longo período de tempo. No IOS, uma CRL armazenada em cache pode ser mantida por até 24 horas após a validade da CRL expirar, e isso pode ser configurado usando este comando na seção de ponto de confiança PKI:

```
crypto pki trustpoint MGMT
 crl cache extend <0 - 1440>
!! here the value is in minutes
```

Em certas circunstâncias, como a emissão de certificados de revogação de CA dentro do período de validade de CRL, o IOS pode ser configurado para excluir o cache com mais frequência. Ao excluir a CRL prematuramente, o IOS é forçado a fazer o download da CRL com mais frequência para manter o cache de CRL atualizado. Esta opção de configuração está disponível na seção de ponto de confiança PKI:

```
crypto pki trustpoint MGMT
 crl cache delete-after <1-43200>
!! here the value is in minutes
```

Por fim, o IOS pode ser configurado para não armazenar em cache o arquivo CRL usando este comando na seção de ponto de confiança PKI:

```
crypto pki trustpoint MGMT
 crl cache none
```

Configuração recomendada

Uma implantação de CA típica com a configuração de CA raiz e sub-CA é como abaixo. O exemplo também inclui uma configuração Sub-CA protegida por um RA.

Com 2048 bits de par de chaves RSA em toda a placa, este exemplo recomenda uma configuração em que:
Root-CA tem uma vida útil de 8 anos
A Sub-CA tem uma vida útil de 3 anos
Os certificados do cliente são emitidos por um ano, que são configurados para solicitar uma renovação de certificado automaticamente.

CA RAIZ - Configuração

```
crypto pki server ROOTCA
database level complete
database archive pkcs12 password p12password
issuer-name CN=RootCA,OU=TAC,O=Cisco
lifetime crl 120
lifetime certificate 1095
lifetime ca-certificate 2920
grant auto rollover ca-cert
auto-rollover 85
database url ftp:://10.1.1.1/CA/ROOT/
database url crl ftp:://10.1.1.1/CA/ROOT/
database url crl publish ftp:://10.1.1.1/WWW/CRL/ROOT/
cdp-url http://10.1.1.1/WWW/CRL/ROOT/ROOTCA.crl
```

SUBCA sem RA - Configuração

```
crypto pki server SUBCA
database level complete
database archive pkcs12 password p12password
issuer-name CN=SubCA,OU=TAC,O=Cisco
lifetime crl 12
lifetime certificate 365
grant auto SUBCA
```

```
auto-rollover 85
database url ftp:://10.1.1.1/CA/SUB/
database url crl ftp:://10.1.1.1/CA/SUB/
database url crl publish ftp:://10.1.1.1/WWW/CRL/SUB/
cdp-url http://10.1.1.1/WWW/CRL/SUB/SUBCA.crl
mode sub-cs


crypto pki trustpoint SUBCA
revocation-check crl
rsakeypair SUBCA 2048
enrollment url http://172.16.1.1
```

SUBCA com RA - Configuração

```
crypto pki server SUBCA
database level complete
database archive pkcs12 password p12password
issuer-name CN=SubCA,OU=TAC,O=Cisco
lifetime crl 12
lifetime certificate 365
grant ra-auto
grant auto rollover ra-cert
auto-rollover 85
 database url ftp:://10.1.1.1/CA/SUB/
database url crl ftp:://10.1.1.1/CA/SUB/
database url crl publish ftp:://10.1.1.1/WWW/CRL/SUB/
cdp-url http://10.1.1.1/WWW/CRL/SUB/SUBCA.crl
mode sub-cs

crypto pki trustpoint SUBCA
revocation-check crl
rsakeypair SUBCA 2048
enrollment url http://172.16.1.1
```

RA para SUBCA - Configuração

```
crypto pki server RA-FOR-SUBCA
database level complete
database archive pkcs12 password p12password
mode ra
grant auto RA-FOR-SUBCA
auto-rollover 85
database url ftp:://10.1.1.1/CA/RA4SUB/


crypto pki trustpoint RA-FOR-SUBCA
enrollment url http://172.16.1.2:80
password ChallengePW123
subject-name CN=RA,OU=ioscs RA,OU=TAC,O=Cisco
revocation-check crl
rsakeypair RA 2048
```

Inscrição de certificado

Inscrição manual

A inscrição manual envolve geração de CSR offline no cliente PKI, que é copiada manualmente

para a CA. O administrador assina manualmente a solicitação, que é importada para o cliente.

Cliente PKI

Configuração do PKI Client:

```
crypto pki trustpoint MGMT
enrollment terminal
serial-number
ip-address none
password ChallengePW123
subject-name CN=PKI-Client,OU=MGMT,OU=TAC,O=Cisco
revocation-check crl
rsakeypair PKI-Key
```

Etapa 1. Primeiro autentique o ponto de confiança (isso também pode ser realizado após a etapa 2).

```
crypto pki authenticate MGMT
!! paste the CA, in this case the SUBCA, certificate in pem format and enter “quit” at the end
in a line by itself]

PKI-Client-1(config)# crypto pki authenticate MGMT

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIDODCCAiCgAwIBAgIBAjANBgkqhkiG9w0BAQUFADAvMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ8wDQYDVQQDEwZSb290Q0EwHhcNMTUxMDE4MjA0MjI3
WhcNMTgxMDE3MjA0MjI3WjAuMQ4wDAYDVQQKEwVDaXNjbzEMMAoGA1UECxMDVEFD
MQ4wDAYDVQQDEwVTdWJDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AJ7hKmBfDo/GOQAEYY/1ptpg28DejUE0ZlDorDkADP2vKfRI0kalSnOs2PIe01ip
7pHFurFVUx/p8teMCkmvnbrSBfyUrWo9YfQeGOELb4d3dSW4jGakm6M8lNRkO7HP
s+IVVTuJSeUZxov6DPa92Y/6HLayX15Iq8ZL+KwmA9oS5NeTi1tBbrcc3Hq8W2Ay
879nDDOqDOsQMQqKtc7E/IA7SBjowImra6FUxzgJ5ye5MymRfRYAH+c4qZJxwHTc
/tSmjiOJlM7X5dtehU/XPEEEbs78peXO9FyzAbhOtCRBVTnhc8WWijq84xu8Oej7
LbXGBKIHSP0uDe32CV0noEUCAwEAAaNgMF4wDwYDVR0TAQH/BAUwAwEB/zALBgNV
HQ8EBAMCAYYwHwYDVR0jBBgwFoAU+oNBdIj9mjpieQ2Z7v79JhKnL68wHQYDVR0O
BBYEFFOv8xtHRojMdJ65oQ2PFBeD5oHiMA0GCSqGSIb3DQEBBQUAA4IBAQAZ/W3P
Wqs4vuQ2jCnVE0v1PVQe/VNS54P/fprQRelceawiBCHA3D0SRgHqUWJUIqBLv4sD
QBegmyTmS76C8YC/jN7VbI30hf6R4qP7CWu8Ef9sWPRC/+Oy6e8AinrK+sVd2dp/
LLDMVoBhS2bQFLWiyRvC9FgyczXRdF+rhKTKeEVXGs7C/Yk/9z+/00rVmSGZAS+v
aPpZWjoC3459t51t8Y3iE6GtjBvmyxBwWt01/5gCu6Mszi7X/kXdmqgNfT5bBBnv
yjWE2ZS8NsH4hwDZpmDJqx4qhrH6bw3iUm+pK9fceZ/HTYasxtcr4NUvvxwXc60y
Wrtlpq3g2XfG+qFB
-----END CERTIFICATE-----
quit
Trustpoint 'MGMT' is a subordinate CA and holds a non self signed cert
Certificate has the following attributes:
      Fingerprint MD5: DBE6AFAC 9E1C3697 01C5466B 78E0DFE3
     Fingerprint SHA1: EAD41B32 BB37BC11 6E0FBC13 41701BFE 200DC46E

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

Etapa 2. Gere a solicitação de assinatura de certificado e leve o CSR para a CA e obtenha o

certificado concedido:

```
PKI-Client-1(config)# crypto pki enroll MGMT
% Start certificate enrollment ..

% The subject name in the certificate will include: CN=PKI-Client,OU=MGMT,OU=TAC,O=Cisco
% The subject name in the certificate will include: PKI-Client-1.cisco.com
% The serial number in the certificate will be: 104Certificate Request follows:

MIIC2zCCAcMCAQAwdTEOMAwGA1UEChMFQ2lzY28xDDAKBgNVBAsTA1RBQzENMAsG
A1UECxMETUdNVDETMBEGA1UEAxMKUEtJLUNsaWVudDExMAoGA1UEBRMDMTA0MCMG
CSqGSIb3DQEJAhYWUEtJLUNsaWVudC0xLmNpc2NvLmNvbTCCASIwDQYJKoZIhvcN
AQEBBQADggEPADCCAQoCggEBANwa7g+DJxG57sMg020w1Fdv9+mIZ6R4livbt7vo
AbW8jpzQlMv4lV3r6ulTJumhBvV7xI+1ZijXP0EqqQZLNboYv37UTJgm83DGO57I
8RTn9DfDQpHiqvhtNuC5S3SCC/hvCxFXnfNXqC3dkfuVkVWoJiLZY87R6j44jUq0
tTL5d8t6lz2L0BeekzKJlOs73gONx0VgQyI/WjDiEwL0xF4DNHURaYyOxBWJc7/B
psDCf7376mb7XXz0LB++E8SvvM/Li6+yQzYv1Lagr0b8C4uE+tCDxG5OniNDiS82
JXsVd43vKRFW85W2ssrElgkuWAvS017XlwK+UDX21dtFdfUCAwEAAaAhMB8GCSqG
SIb3DQEJDjESMBAwDgYDVR0PAQH/BAQDAgWgMA0GCSqGSIb3DQEBBQUAA4IBAQA+
UqkqUZZar9TdmB8I7AHku5m79l42o8cuhwOccehxE6jmzh9P+Ttb9Me7l7L8Y2iR
yYyJHsL7m6tjK2+Gllg7RJdoxG8l8aMZS1ruXOBqFBrmo7OSz1nfXpiTyh88jyca
Hw/8G8uaYuQbZIj53BWmQGRpm7J//ktn0D4W3Euh9HttMuYYX7BOct05BLqqiCCw
n+kKHZxzGXy7JSZpUlDtvPPnnuqWK7iVoy3vtV6GoFOrxRoo05QVFehS0/m4NFQI
mXA0eTEgujSaQi4iWte/UxruO/3p/eHr67MtZXLRl0YDFgaQd7vD7fCsDx5pquKV
jNEUT6FNHdsnqrAKqodO

---End - This line not part of the certificate request---

Redisplay enrollment request? [yes/no]: no
```

Etapa 3. Agora importe o certificado concedido via terminal:

```
PKI-Client-1(config)# crypto pki import MGMT certificate

Enter the base 64 encoded certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIDcDCCAligAwIBAgIBAzANBgkqhkiG9w0BAQQFADAuMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ4wDAYDVQQDEwVTdWJDQTAeFw0xNTEwMTkyMDM1MDZa
Fw0xNjEwMTgyMDM1MDZaMHUxDjAMBgNVBAoTBUNpc2NvMQwwCgYDVQQLEwNUQUMx
DTALBgNVBAsTBE1HTVQxEzARBgNVBAMTClBLSS1DbGllbnQxMTAKBgNVBAUTAzEw
NDAjBgkqhkiG9w0BCQIWFlBLSS1DbGllbnQtMS5jaXNjby5jb20wggEiMA0GCSqG
SIb3DQEBAQUAA4IBDwAwggEKAoIBAQDcGu4PgycRue7DINNtMNRXb/fpiGekeJYr
27e76AG1vI6c0JTL+JVd6+rpUybpoQb1e8SPtWYo1z9BKqkGSzW6GL9+1EyYJvNw
xjueyPEU5/Q3w0KR4qr4bTbguUt0ggv4bwsRV53zV6gt3ZH7lZFVqCYi2WPO0eo+
OI1KtLUy+XfLepc9i9AXnpMyiZTrO94DjcdFYEMiP1ow4hMC9MReAzR1EWmMjsQV
iXO/wabAwn+9++pm+1189CwfvhPEr7zPy4uvskM2L9S2oK9G/AuLhPrQg8RuTp4j
Q4kvNiV7FXeN7ykRVvOVtrLKxJYJLlgL0tNe15cCvlA19tXbRXX1AgMBAAGjUjBQ
MA4GA1UdDwEB/wQEAwIFoDAfBgNVHSMEGDAWgBRTr/MbR0aIzHSeuaENjxQXg+aB
4jAdBgNVHQ4EFgQUK+9/lrlL+TyYxvsgxzPwwrhmS5UwDQYJKoZIhvcNAQEEBQAD
ggEBAIrLrzFLnm9z7ula1uRh03r6dSCFy9XkOk6ZaHfksbENoDmkcgIwKoAsSF9E
rQmA9W5qXVU7PEsqOmcu8zEv7uuiqM4D4nDP69HsyToPjxVcoG7PSyKJYnXRgkVa
IYyMaSaRKWlhb2uWj3XPLzS0/ZBOGAG9rMBVzaqLfLAZgnQUVJvwsNofe+ASojk9
mCRsEHD8WVuAzcnwYKXx3j3x/T7jbB3ibPfbYKQqlS12XFHhJoK+HfSA2fyZBFLF
syN/B2Ow0bvc71YlYOQuYwz3XOMIHD6vARTO4f0ZIQti2dy1kHc+5lIdhLsn/bA5
yUo7WxnAE8LOoYIf9iU9q0mqkMU=
-----END CERTIFICATE-----
```

```
quit
% Router Certificate successfully imported
```

Servidor PKI

Etapa 1. Primeiro, exporte o certificado CA de emissão da CA, que, neste caso, é o certificado SUBCA. Isso é importado durante a etapa 1 acima no cliente PKI, ou seja, autenticação de ponto confiável.

```
SUBCA(config)# crypto pki export SUBCA pem terminal
% CA certificate: !! Root-CA certificate
-----BEGIN CERTIFICATE-----
MIIDPDCCAiSgAwIBAgIBATANBgkqhkiG9w0BAQQFADAvMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ8wDQYDVQQDEwZSb290Q0EwHhcNMTUxMDE4MjAwOTIx
WhcNMjMxMDE2MjAwOTIxWjAvMQ4wDAYDVQQKEwVDaXNjbzEMMAoGA1UECxMDVEFD
MQ8wDQYDVQQDEwZSb290Q0EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQCajfMy8gU3ZXQfKgP/wYKLB0cuywzYcDaSoNVlEvUZOWgUltCGP4CiCXyw0U0U
Zmy0rusibMV7mtkTX5muaPC0XfT98rswPiZV0qvEYpHF2YodPOUoqR3FeKj/tDbI
IikcLrfj87aeMJjCrWD888wfTN9Hw9x2QVDoSxLbzTLticXdXxwS5wxlM16GspmT
WL4fg1JRWgjRqMmOcpf716Or88XJ2N2HeWxxVFIwYQf3thHR6DgTdCgJ1uqjVE6q
1LQ1g8k81mvuCZX0uLZiTMJ69xo+Ot/RpeeE2RShxK5rh56ObQq4MT4lbIPKqIxU
lbKzWdh10NiYwjgTNwTs9GGvAgMBAAGjYzBhMA8GA1UdEwEB/wQFMAMBAf8wDgYD
VR0PAQH/BAQDAgGGMB8GA1UdIwQYMBaAFPqDQXSI/Zo6YnkNme7+/SYSpy+vMB0G
A1UdDgQWBBT6g0F0iP2aOmJ5DZnu/v0mEqcvrzANBgkqhkiG9w0BAQQFAAOCAQEA
VKwqI9vpmoRh9QoOJGtOA3qEgV4eCfXdMuYxmmo0sdaBYBfQm2RhZeQ1X90vVBso
G4Wx6cJVSXCtkqZTm1IoMtya+gdhLbKqZmxc+I5/js88SrbrBIm4zj+sOoySV9kW
THEEmZjdTCWXo2wNCr23gGdnb4RqZ0FTOfoZO/2Xnpcbvhz2/K7wlDRJ5k1wrsRW
RRwsQEh4LYMFIg0aBs4gmRLZ8ytwrvvrhQTVrAA/MeomUEPhcIYESg1AlWxoCYZU
0iqKfDa9+4wEJ+PMGDhM2UV0fuP0rWitKWxecSVbo54z3VHYwwCbz2jCs8XGE61S
+XlxCZKFVdlVaMmuaZTdFg==
-----END CERTIFICATE-----

% General Purpose Certificate: !! SUBCA certificate
-----BEGIN CERTIFICATE-----
MIIDODCCAiCgAwIBAgIBAjANBgkqhkiG9w0BAQUFADAvMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ8wDQYDVQQDEwZSb290Q0EwHhcNMTUxMDE4MjA0MjI3
WhcNMTgxMDE3MjA0MjI3WjAuMQ4wDAYDVQQKEwVDaXNjbzEMMAoGA1UECxMDVEFD
MQ4wDAYDVQQDEwVTdWJDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AJ7hKmBfDo/GOQAEYY/1ptpg28DejUE0ZlDorDkADP2vKfRI0kalSnOs2PIe01ip
7pHFurFVUx/p8teMCkmvnbrSBfyUrWo9YfQeGOELb4d3dSW4jGakm6M8lNRkO7HP
s+IVVTuJSeUZxov6DPa92Y/6HLayX15Iq8ZL+KwmA9oS5NeTi1tBbrcc3Hq8W2Ay
879nDDOqDOsQMQqKtc7E/IA7SBjowImra6FUxzgJ5ye5MymRfRYAH+c4qZJxwHTc
/tSmjiOJlM7X5dtehU/XPEEEbs78peXO9FyzAbhOtCRBVTnhc8WWijq84xu8Oej7
LbXGBKIHSP0uDe32CV0noEUCAwEAAaNgMF4wDwYDVR0TAQH/BAUwAwEB/zALBgNV
HQ8EBAMCAYYwHwYDVR0jBBgwFoAU+oNBdIj9mjpieQ2Z7v79JhKnL68wHQYDVR0O
BBYEFFOv8xtHRojMdJ65oQ2PFBeD5oHiMA0GCSqGSIb3DQEBBQUAA4IBAQAZ/W3P
Wqs4vuQ2jCnVE0v1PVQe/VNS54P/fprQRelceawiBCHA3D0SRgHqUWJUIqBLv4sD
QBegmyTmS76C8YC/jN7VbI30hf6R4qP7CWu8Ef9sWPRC/+Oy6e8AinrK+sVd2dp/
LLDMVoBhS2bQFLWiyRvC9FgyczXRdF+rhKTKeEVXGs7C/Yk/9z+/00rVmSGZAS+v
aPpZWjoC3459t51t8Y3iE6GtjBvmyxBwWt01/5gCu6Mszi7X/kXdmqgNfT5bBBnv
yjWE2ZS8NsH4hwDZpmDJqx4qhrH6bw3iUm+pK9fceZ/HTYasxtcr4NUvvxwXc60y
Wrtlpq3g2XfG+qFB
-----END CERTIFICATE-----
```

Etapa 2. Depois da Etapa 2 no PKI-Client, pegue o CSR do cliente e forneça-o para fazer a assinatura na SUBCA usando este comando:

```
crypto pki server SUBCA request pkcs10 terminal pem
```

Esse comando sugere que a SUBCA aceita uma solicitação de assinatura de certificado do

terminal e, uma vez concedida, os dados do certificado são impressos no formato PEM.

```
SUBCA# crypto pki server SUBCA request pkcs10 terminal pem
PKCS10 request in base64 or pem

% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
MIIC2zCCAcMCAQAwdTEOMAwGA1UEChMFQ2lzY28xDDAKBgNVBAsTA1RBQzENMAsG
A1UECxMETUdNVDETMBEGA1UEAxMKUEtJLUNsaWVudDExMAoGA1UEBRMDMTA0MCMG
CSqGSIb3DQEJAhYWUEtJLUNsaWVudC0xLmNpc2NvLmNvbTCCASIwDQYJKoZIhvcN
AQEBBQADggEPADCCAQoCggEBANwa7g+DJxG57sMg020w1Fdv9+mIZ6R4livbt7vo
AbW8jpzQlMv4lV3r6ulTJumhBvV7xI+1ZijXP0EqqQZLNboYv37UTJgm83DGO57I
8RTn9DfDQpHiqvhtNuC5S3SCC/hvCxFXnfNXqC3dkfuVkVWoJiLZY87R6j44jUq0
tTL5d8t6lz2L0BeekzKJlOs73gONx0VgQyI/WjDiEwL0xF4DNHURaYyOxBWJc7/B
psDCf7376mb7XXz0LB++E8SvvM/Li6+yQzYv1Lagr0b8C4uE+tCDxG5OniNDiS82
JXsVd43vKRFW85W2ssrElgkuWAvS017XlwK+UDX21dtFdfUCAwEAAaAhMB8GCSqG
SIb3DQEJDjESMBAwDgYDVR0PAQH/BAQDAgWgMA0GCSqGSIb3DQEBBQUAA4IBAQA+
UqkqUZZar9TdmB8I7AHku5m79l42o8cuhwOccehxE6jmzh9P+Ttb9Me7l7L8Y2iR
yYyJHsL7m6tjK2+Gllg7RJdoxG8l8aMZS1ruXOBqFBrmo7OSz1nfXpiTyh88jyca
Hw/8G8uaYuQbZIj53BWmQGRpm7J//ktn0D4W3Euh9HttMuYYX7BOct05BLqqiCCw
n+kKHZxzGXy7JSZpUlDtvPPnnuqWK7iVoy3vtV6GoFOrxRoo05QVFehS0/m4NFQI
mXA0eTEgujSaQi4iWte/UxruO/3p/eHr67MtZXLRl0YDFgaQd7vD7fCsDx5pquKV
jNEUT6FNHdsnqrAKqodO
quit
% Enrollment request pending, reqId=1
```

Se a CA estiver no modo de concessão automática, o certificado concedido será exibido no formato PEM acima. Quando a CA está no modo de concessão manual, a solicitação de certificado está marcada como **pendente**, recebe um valor de id e está na fila de solicitação de inscrição.

```
SUBCA#show crypto pki server SUBCA requests
Enrollment Request Database:

Router certificates requests:
ReqID  State      Fingerprint                      SubjectName
--------------------------------------------------------------
1      pending    7710276982EA176324393D863C9E350E serialNumber=104+hostname=PKI-Client-
1.cisco.com,cn=PKI-Client,ou=MGMT,ou=TAC,o=Cisco
```

Etapa 3. Conceda manualmente esta solicitação usando este comando:

```
SUBCA# crypto pki server SUBCA grant 1
% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIDcDCCAligAwIBAgIBAzANBgkqhkiG9w0BAQQFADAuMQ4wDAYDVQQKEwVDaXNj
bzEMMAoGA1UECxMDVEFDMQ4wDAYDVQQDEwVTdWJDQTAeFw0xNTEwMTkyMDM1MDZa
Fw0xNjEwMTgyMDM1MDZaMHUxDjAMBgNVBAoTBUNpc2NvMQwwCgYDVQQLEwNUQUMx
DTALBgNVBAsTBE1HTVQxEzARBgNVBAMTClBLSS1DbGllbnQxMTAKBgNVBAUTAzEw
NDAjBgkqhkiG9w0BCQIWFlBLSS1DbGllbnQtMS5jaXNjby5jb20wggEiMA0GCSqG
SIb3DQEBAQUAA4IBDwAwggEKAoIBAQDcGu4PgycRue7DINNtMNRXb/fpiGekeJYr
27e76AG1vI6c0JTL+JVd6+rpUybpoQb1e8SPtWYo1z9BKqkGSzW6GL9+1EyYJvNw
xjueyPEU5/Q3w0KR4qr4bTbguUt0ggv4bwsRV53zV6gt3ZH7lZFVqCYi2WPO0eo+
OI1KtLUy+XfLepc9i9AXnpMyiZTrO94DjcdFYEMiP1ow4hMC9MReAzR1EWmMjsQV
iXO/wabAwn+9++pm+1189CwfvhPEr7zPy4uvskM2L9S2oK9G/AuLhPrQg8RuTp4j
Q4kvNiV7FXeN7ykRVvOVtrLKxJYJLlgL0tNe15cCvlA19tXbRXX1AgMBAAGjUjBQ
MA4GA1UdDwEB/wQEAwIFoDAfBgNVHSMEGDAWgBRTr/MbR0aIzHSeuaENjxQXg+aB
```

```
4jAdBgNVHQ4EFgQUK+9/lrlL+TyYxvsgxzPwwrhmS5UwDQYJKoZIhvcNAQEEBQAD
ggEBAIrLrzFLnm9z7ula1uRh03r6dSCFy9XkOk6ZaHfksbENoDmkcgIwKoAsSF9E
rQmA9W5qXVU7PEsqOmcu8zEv7uuiqM4D4nDP69HsyToPjxVcoG7PSyKJYnXRgkVa
IYyMaSaRKWlhb2uWj3XPLzS0/ZBOGAG9rMBVzaqLfLAZgnQUVJvwsNofe+ASojk9
mCRsEHD8WVuAzcnwYKXx3j3x/T7jbB3ibPfbYKQqlS12XFHhJoK+HfSA2fyZBFLF
syN/B2Ow0bvc71YlYOQuYwz3XOMIHD6vARTO4f0ZIQti2dy1kHc+5lIdhLsn/bA5
yUo7WxnAE8LOoYIf9iU9q0mqkMU=
-----END CERTIFICATE-----
```

Note: A inscrição manual de uma Sub-CA em uma CA raiz não é possível.

Note: Uma AC em um estado desabilitado devido ao servidor HTTP desabilitado pode conceder manualmente as solicitações de certificado.

Inscrição usando SCEP

A configuração do PKI Client é:

```
crypto pki trustpoint MGMT
enrollment url http://172.16.1.2:80
 serial-number
 ip-address none
 password 7 110A1016141D5A5E57
 subject-name CN=PKI-Client,OU=MGMT,OU=TAC,O=Cisco
 revocation-check crl
 rsakeypair PKI-Key 2048
```

A configuração do servidor PKI é:

```
SUBCA# show run all | section pki server
crypto pki server SUBCA
 database level complete
 database archive pkcs12 password 7 01100F175804575D72
 issuer-name CN=SubCA,OU=TAC,O=Cisco
 lifetime crl 12
 lifetime certificate 365
 lifetime ca-certificate 1095
 lifetime enrollment-request 168
 mode sub-cs
 auto-rollover 85
 database url ftp:://10.1.1.1/CA/SUB/
database url crl ftp:://10.1.1.1/CA/SUB/
database url crl publish ftp:://10.1.1.1/WWW/CRL/SUB/
```

O modo padrão de concessão de solicitação de certificado é manual:

```
SUBCA# show crypto pki server
Certificate Server SUBCA:
   Status: enabled
   State: enabled
   Server's configuration is locked  (enter "shut" to unlock it)
   Issuer name: CN=SubCA,OU=TAC,O=Cisco
   CA cert fingerprint: DBE6AFAC 9E1C3697 01C5466B 78E0DFE3
   Server configured in subordinate server mode
```

```
Upper CA cert fingerprint: CD0DE4C7 955EFD60 296B7204 41FB6EF6
 Granting mode is: manual
Last certificate issued serial number (hex): 4
CA certificate expiration timer: 21:42:27 CET Oct 17 2018
CRL NextUpdate timer: 09:42:37 CET Oct 20 2015
Current primary storage dir: unix:/SUB/
Current storage dir for .crl files: unix:/SUB/
Database Level: Complete - all issued certs written as <serialnum>.cer
Auto-Rollover configured, overlap period 85 days
Autorollover timer: 21:42:27 CET Jul 24 2018
```

Concessão manual

Etapa 1. Cliente PKI: Como primeiro passo, que é obrigatório, autentique o ponto de confiança no cliente PKI:

```
PKI-Client-1(config)# crypto pki authenticate MGMT
Trustpoint 'MGMT' is a subordinate CA and holds a non self signed cert
Certificate has the following attributes:
      Fingerprint MD5: DBE6AFAC 9E1C3697 01C5466B 78E0DFE3
     Fingerprint SHA1: EAD41B32 BB37BC11 6E0FBC13 41701BFE 200DC46E

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
```

Etapa 2. Cliente PKI: Após a autenticação do ponto de confiança, o cliente PKI pode ser inscrito para um certificado.

> **Note**: Se a inscrição automática estiver configurada, o cliente executará automaticamente a inscrição.

```
config terminal
crypto pki enroll MGMT
```

Nos bastidores, esses eventos acontecem:

- O IOS procura um par de chaves RSA chamado PKI-Key. Se existir, ele é coletado para solicitar um certificado de identidade. Caso contrário, o IOS cria um par de chaves de 2048 bits chamado PKI-Key e o usa para solicitar um certificado de identidade.
- O IOS cria uma solicitação de assinatura de certificado no formato PKCS10.
- Em seguida, o IOS criptografa esse CSR usando uma chave simétrica aleatória. A chave simétrica aleatória é criptografada usando a chave pública do destinatário, que é a SUBCA (a chave pública da SUBCA está disponível devido à autenticação de ponto de confiança). O CSR criptografado, a chave simétrica aleatória criptografada e as informações do destinatário são reunidas em dados com envelope PKCS#7.
- Esses dados com envelope PKCS#7 são assinados usando um certificado autoassinado temporário durante a inscrição inicial. Os dados com envelope PKCS#7, o certificado de assinatura usado pelo cliente e a assinatura do cliente são reunidos em um pacote de dados assinado PKCS#7. Esta é a base64 codificada e, em seguida, a URL codificada. O blob de dados resultante é enviado como argumento de "mensagem" no URI HTTP enviado à CA:

```
GET /cgi-bin/pkiclient.exe?operation=PKIOperation&message=MII…. HTTP/1.0
```

Etapa 3. Servidor PKI:

Quando o IOS PKI Server recebe a solicitação, ele verifica:

1. Verifica se o banco de dados de solicitação de inscrição contém uma solicitação de certificado com a mesma id de transação associada à nova solicitação.

> **Note**: Uma ID de transação é um hash MD5 da chave pública, para o qual um certificado de identidade está sendo solicitado pelo cliente.

2. Verifica se o banco de dados de solicitação de inscrição contém uma solicitação de certificado com a mesma senha de desafio que a enviada pelo cliente.

> **Note**: Se (1) retornar true ou ambos (1) e (2) retornarem true, então um servidor CA pode rejeitar a solicitação com base em uma solicitação de identidade duplicada. No entanto, nesse caso, o IOS PKI Server substitui a solicitação mais antiga pela solicitação mais nova.

Etapa 4. Servidor PKI:

Conceda manualmente as solicitações no servidor PKI:

Para exibir a solicitação:

```
show crypto pki server SUBCA requests
```

Para conceder uma solicitação específica ou todas as solicitações:

```
crypto pki server SUBCA grant <id|all>
```

Etapa 5. Cliente PKI:

Enquanto isso, um cliente PKI inicia um temporizador de POLL. Aqui, o IOS executa GetCertInitial em intervalos regulares até que SCEP CertRep = CONCEDIDO junto com o certificado concedido seja recebido pelo cliente.

Quando o certificado concedido é recebido, o IOS o instala automaticamente.