Public Key Infrastructure Configuration Guide, Cisco IOS XE Gibraltar 16.11.x

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

CONTENTS

CHAPTER 1

Read Me First

Important Information about Cisco IOS XE 16

Effective Cisco IOS XE Release 3.7.0E for Catalyst Switching and Cisco IOS XE Release 3.17S (for Access and Edge Routing) the two releases evolve (merge) into a single version of converged release—the Cisco IOS XE 16—providing one release covering the extensive range of access and edge products in the Switching and Routing portfolio.

Feature Information

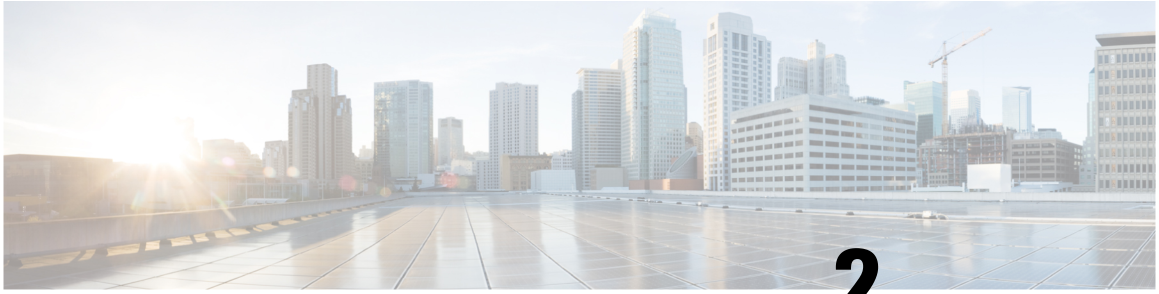
Use Cisco Feature Navigator to find information about feature support, platform support, and Cisco software image support. An account on Cisco.com is not required.

Related References

- Cisco IOS Command References, All Releases

Obtaining Documentation and Submitting a Service Request

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.



CHAPTER 2

Cisco IOS XE PKI Overview

Cisco IOS XE public key infrastructure (PKI) provides certificate management to support security protocols such as IP Security (IPSec), secure shell (SSH), and secure socket layer (SSL).

This module identifies and describes concepts that are needed to understand, plan for, and implement a PKI.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Cisco IOS XE PKI

What Is Cisco IOS XE PKI

A PKI is composed of the following entities:

- Peers communicating on a secure network
- At least one certification authority (CA) that grants and maintains certificates
- Digital certificates, which contain information such as the certificate validity period, peer identity information, encryptions keys that are used for secure communications, and the signature of the issuing CA

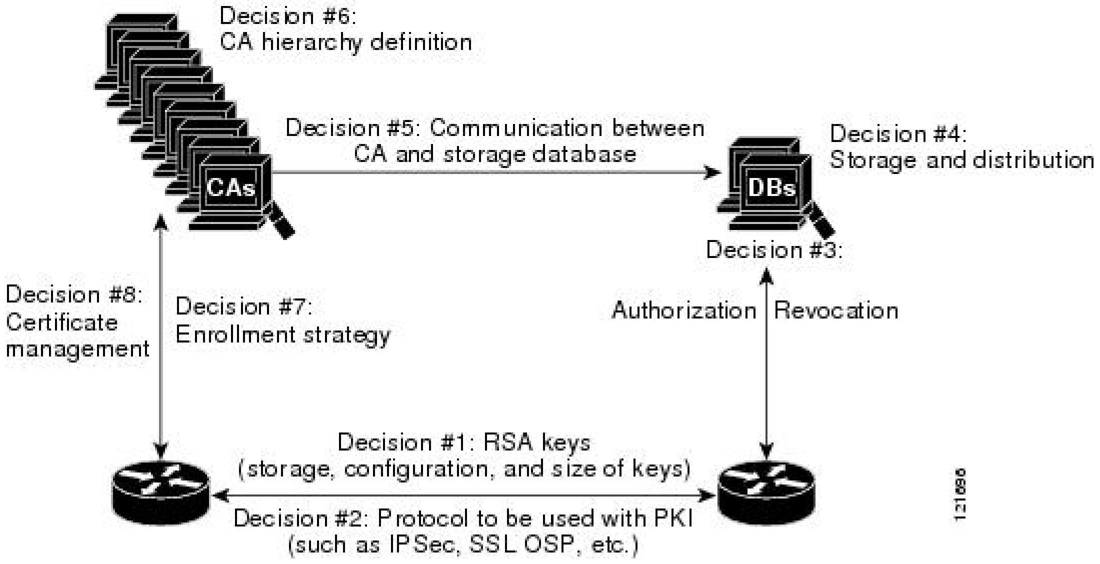
- An optional registration authority (RA) to offload the CA by processing enrollment requests
- A distribution mechanism (such as Lightweight Directory Access Protocol [LDAP] or HTTP) for certificate revocation lists (CRLs)

PKI provides customers with a scalable, secure mechanism for distributing, managing, and revoking encryption and identity information in a secured data network. Every entity (a person or a device) participating in the secured communicated is enrolled in the PKI in a process where the entity generates an Rivest, Shamir, and Adelman (RSA) key pair (one private key and one public key) and has their identity validated by a trusted entity (also known as a CA or trustpoint).

After each entity enrolls in a PKI, every peer (also known as an end host) in a PKI is granted a digital certificate that has been issued by a CA. When peers must negotiate a secured communication session, they exchange digital certificates. Based on the information in the certificate, a peer can validate the identity of another peer and establish an encrypted session with the public keys contained in the certificate.

Although you can plan for and set up your PKI in a number of different ways, the figure below shows the major components that make up a PKI and suggests an order in which each decision within a PKI can be made. The figure is a suggested approach; you can choose to set up your PKI from a different perspective.

Figure 1: Deciding How to Set Up Your PKI


Decision #6:
CA hierarchy definition
Decision #5: Communication between CA and storage database
Decision #4:
Storage and distribution
CAs
DBs
Decision #3:
Authorization
Revocation
Decision #8:
Certificate management
Decision #7:
Enrollment strategy
Decision #1: RSA keys (storage, configuration, and size of keys)
Decision #2: Protocol to be used with PKI (such as IPSec, SSL OSP, etc.)
121896

RSA Keys Overview

An RSA key pair consists of a public key and a private key. When setting up your PKI, you must include the public key in the certificate enrollment request. After the certificate has been granted, the public key will be included in the certificate so that peers can use it to encrypt data that is sent to the router. The private key is kept on the router and used both to decrypt the data sent by peers and to digitally sign transactions when negotiating with peers.

RSA key pairs contain a key modulus value. The modulus determines the size of the RSA key. The larger the modulus, the more secure the RSA key. However, keys with large modulus values take longer to generate, and encryption and decryption operations take longer with larger keys.

Note The default key size is 1024 bit.

What Are CAs

A CA, also known as a trustpoint, manages certificate requests and issues certificates to participating network devices. These services (managing certificate requests and issuing certificates) provide centralized key management for the participating devices and are explicitly trusted by the receiver to validate identities and to create digital certificates. Before any PKI operations can begin, the CA generates its own public key pair and creates a self-signed CA certificate; thereafter, the CA can sign certificate requests and begin peer enrollment for the PKI.

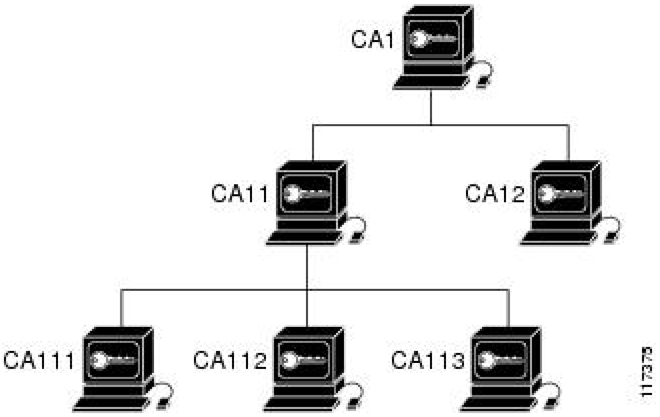
You can use a CA provided by a third-party CA vendor, or you can use an "internal" CA, which is the Cisco IOS Certificate Server.

Hierarchical PKI Multiple CAs

PKI can be set up in a hierarchical framework to support multiple CAs. At the top of the hierarchy is a root CA, which holds a self-signed certificate. The trust within the entire hierarchy is derived from the RSA key pair of the root CA. The subordinate CAs within the hierarchy can be enrolled with either the root CA or with another subordinate CA. These enrollment options are how multiple tiers of CAs are configured. Within a hierarchical PKI, all enrolled peers, can validate the certificate of one another if the peers share a trusted root CA certificate or a common subordinate CA.

The figure below shows the enrollment relationships among CAs within a three-tiered hierarchy.

Figure 2: Three-Tiered CA Hierarchy Sample Topology



Each CA corresponds to a trustpoint. For example, CA11 and CA12 are subordinate CAs, holding CA certificates that have been issued by CA1; CA111, CA112, and CA113 are also subordinate CAs, but their CA certificates have been issued by CA11.

When to Use Multiple CAs

Multiple CAs provide users with added flexibility and reliability. For example, subordinate CAs can be placed in branch offices while the root CA is at the office headquarters. Also, different granting policies can be implemented per CA, so you can set up one CA to automatically grant certificate requests while another CA within the hierarchy requires each certificate request to be manually granted.

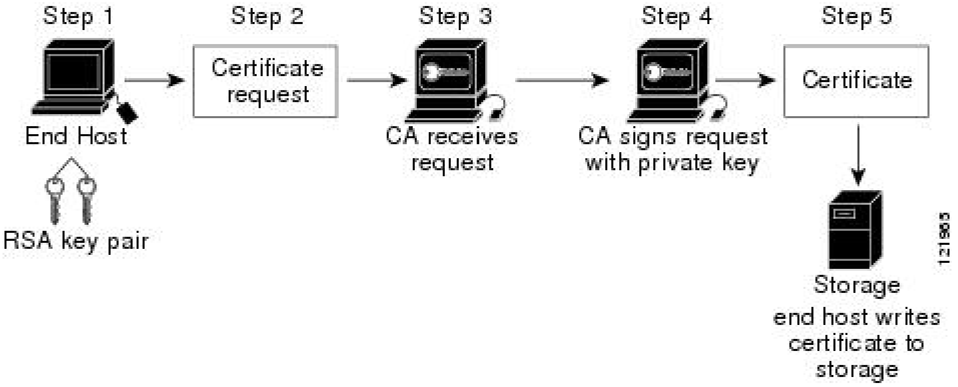
Scenarios in which at least a two-tier CA is recommended are as follows:

- Large and very active networks in which a large number of certificates are revoked and reissued. A multiple tier CA helps to control the size of the CRLs.
- When online enrollment protocols are used, the root CA can be kept offline with the exception of issuing subordinate CA certificates. This scenario provides added security for the root CA.

Certificate Enrollment How It Works

Certificate enrollment is the process of obtaining a certificate from a CA. Each end host that wants to participate in the PKI must obtain a certificate. Certificate enrollment occurs between the end host requesting the certificate and the CA. The table below and the following steps describe the certificate enrollment process.

Figure 3: Certificate Enrollment Process



1. The end host generates an RSA key pair.
2. The end host generates a certificate request and forwards it to the CA (or the RA, if applicable).
3. The CA receives the certificate enrollment request, and, depending on your network configuration, one of the following options occurs:
   a. Manual intervention is required to approve the request.
   b. The end host is configured to automatically request a certificate from the CA. Thus, operator intervention is no longer required at the time the enrollment request is sent to the CA server.

Note If you configure the end host to automatically request certificates from the CA, you should have an additional authorization mechanism.

1. After the request is approved, the CA signs the request with its private key and returns the completed certificate to the end host.
2. The end host writes the certificate to a storage area such as NVRAM.

Certificate Enrollment Via Secure Device Provisioning

Secure Device Provisioning (SDP) is a web-based certificate enrollment interface that can be used to easily deploy PKI between two end devices, such as a Cisco IOS XE client and a Cisco IOS certificate server.

SDP (also refer red to as Trusted Transitive Introduction [TTI]) is a communication protocol that provides a bidirectional introduction between two end entities, such as a new network device and a VPN. SDP involves the following three entities:

- Introducer—A mutually trusted device that introduces the petitioner to the registrar. The introducer can be a device user, such as a system administrator.
- Petitioner—A new device that is joined to the secure domain.
- Registrar—A certificate server or other server that authorizes the petitioner.

SDP is implemented over a web browser in three phases—welcome, introduction, and completion. Each phase is shown to the user via a web page.

Certificate Revocation Why It Occurs

After each participant has successfully enrolled in the PKI, the peers are ready to begin negotiations for a secure connection with each other. Thus, the peers present their certificates for validation followed by a revocation check. After the peer verifies that the other peer’s certificate was issued by an authenticated CA, the CRL or Online Certificate Status Protocol (OCSP) server is checked to ensure that the certificate has not been revoked by the issuing CA. The certificate usually contains a certificate distribution point (CDP) in the form of a URL. Cisco IOS software uses the CDP to locate and retrieve the CRL. If the CDP server does not respond, the Cisco IOS software reports an error, which may result in the peer’s certificate being rejected.

Planning for a PKI

Planning for a PKI requires evaluating the requirements and expected use for each of the PKI components. It is recommended that you (or the network administrator) thoroughly plan the PKI before beginning any PKI configuration.

Although there are a number of approaches to consider when planning the PKI, this document begins with peer-to-peer communication. However you or the network administrator choose to plan the PKI, understand that certain decisions influence other decisions within the PKI. For example, the enrollment and deployment strategy could influence the planned CA hierarchy. Thus, it is important to understand how each component functions within the PKI and how certain component options are dependent upon decisions made earlier in the planning process.

Where to Go Next

After you have generated an RSA key pair, you should set up the trustpoint. If you have already set up the trustpoint, you should authenticate and enroll the routers in a PKI. For information on enrollment, see the module “Configuring Certificate Enrollment for a PKI.”

Additional References for Understanding and Planning a PKI

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
PKI and security commands	• *Cisco IOS Security Command Reference Commands A to C* • *Cisco IOS Security Command Reference Commands D to L* • *Cisco IOS Security Command Reference Commands M to R* • *Cisco IOS Security Command Reference Commands S to Z*
USB Token RSA Operations: Using the RSA keys on a USB token for initial autoenrollment	*Configuring Certificate Enrollment for a PKI*
USB Token RSA Operations: Benefits of using USB tokens	*Storing PKI Credentials*
Certificate server client certificate enrollment, autoenrollment, and automatic rollover	*Configuring Certificate Enrollment for a PKI*
Setting up and logging into a USB token	*Storing PKI Credentials*
Web-based certificate enrollment	*Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI*
RSA keys in PEM formatted files	*Deploying RSA Keys Within a PKI*
Choosing a certificate revocation mechanism	*Configuring Authorization and Revocation of Certificates in a PKI*
Recommended cryptographic algorithms	*Next Generation Encryption*

MIBs

MIBs	MIBs Link
• PKI MIB	To locate and download MIBs for selected platforms, Cisco IOS software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Glossary

CDP—certificate distribution point. Field within a digital certificate containing information that describes how to retrieve the CRL for the certificate. The most common CDPs are HTTP and LDAP URLs. A CDP may also contain other types of URLs or an LDAP directory specification. Each CDP contains one URL or directory specification.

certificates—Electronic documents that bind a user's or device's name to its public key. Certificates are commonly used to validate a digital signature.

CRL—certificate revocation list. Electronic document that contains a list of revoked certificates. The CRL is created and digitally signed by the CA that originally issued the certificates. The CRL contains dates for when the certificate was issued and when it expires. A new CRL is issued when the current CRL expires.

CA—certification authority. Service responsible for managing certificate requests and issuing certificates to participating IPSec network devices. This service provides centralized key management for the participating devices and is explicitly trusted by the receiver to validate identities and to create digital certificates.

peer certificate--Certificate presented by a peer, which contains the peer's public key and is signed by the trustpoint CA.

PKI—public key infrastructure. System that manages encryption keys and identity information for components of a network that participate in secured communications.

RA—registration authority. Server that acts as a proxy for the CA so that CA functions can continue when the CA is offline. Although the RA is often part of the CA server, the RA could also be an additional application, requiring an additional device to run it.

RSA keys—Public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adleman. An RSA key pair (a public and a private key) is required before you can obtain a certificate for your router.

CHAPTER 3

Deploying RSA Keys Within a PKI

This module explains how to set up and deploy Rivest, Shamir, and Adelman (RSA) keys within a public key infrastructure (PKI). An RSA key pair (a public and a private key) is required before you can obtain a certificate for your router; that is, the end host must generate a pair of RSA keys and exchange the public key with the certification authority (CA) to obtain a certificate and enroll in a PKI.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring RSA Keys for a PKI

- Before setting up and deploying RSA keys for a PKI, you should be familiar with the module Cisco IOS PKI Overview: Understanding and Planning a PKI .

Information About RSA Keys Configuration

RSA Keys Overview

An RSA key pair consists of a public key and a private key. When setting up your PKI, you must include the public key in the certificate enrollment request. After the certificate has been granted, the public key will be included in the certificate so that peers can use it to encrypt data that is sent to the router. The private key is kept on the router and used both to decrypt the data sent by peers and to digitally sign transactions when negotiating with peers.

RSA key pairs contain a key modulus value. The modulus determines the size of the RSA key. The larger the modulus, the more secure the RSA key. However, keys with large modulus values take longer to generate, and encryption and decryption operations take longer with larger keys.

Usage RSA Keys Versus General-Purpose RSA Keys

There are two mutually exclusive types of RSA key pairs--usage keys and general-purpose keys. When you generate RSA key pairs (via the **crypto key generate rsa** command), you will be prompted to select either usage keys or general-purpose keys.

Usage RSA Keys

Usage keys consist of two RSA key pairs--one RSA key pair is generated and used for encryption and one RSA key pair is generated and used for signatures. With usage keys, each key is not unnecessarily exposed. (Without usage keys, one key is used for both authentication methods, increasing the exposure of that key.)

General-Purpose RSA Keys

General-purpose keys consist of only one RSA key pair that used for both encryption and signatures. General-purpose key pairs are used more frequently than usage key pairs.

How RSA Key Pairs are Associated with a Trustpoint

A trustpoint, also known as the certificate authority (CA), manages certificate requests and issues certificates to participating network devices. These services provide centralized key management for the participating devices and are explicitly trusted by the receiver to validate identities and to create digital certificates. Before any PKI operations can begin, the CA generates its own public key pair and creates a self-signed CA certificate; thereafter, the CA can sign certificate requests and begin peer enrollment for the PKI.

Caution Do not manually generate an rsa keypair under trustpoint. If we want to manually generate the keys, generate the key pairs as usage-keys and not as general-purpose keys.

Caution Certificate renewal with regenerate option does not work with key label starting from zero ('0'), (for example, '0test'). CLI allows configuring such name under trustpoint, and allows hostname starting from zero. When configuring **rsakeypair** *name* under a trustpoint, do not configure the name starting from zero. When keypair name is not configured and the default keypair is used, make sure the router hostname does not start from zero. If it does so, configure "**rsakeypair** *name* explicitly under the trustpoint with a different name.

Reasons to Store Multiple RSA Keys on a Router

Configuring multiple RSA key pairs allows the Cisco IOS software to maintain a different key pair for each CA with which it is dealing or the software can maintain multiple key pairs and certificates with the same CA. As a result, the Cisco IOS software can match policy requirements for each CA without compromising the requirements specified by the other CAs, such as key length, key lifetime, and general-purpose versus usage keys.

Named key pairs (which are specified via the **label** *key-label* option) allow you to have multiple RSA key pairs, enabling the Cisco IOS software to maintain a different key pair for each identity certificate.

Benefits of Exportable RSA Keys

Caution Exportable RSA keys should be carefully evaluated before use because using exportable RSA keys introduces the risk that these keys might be exposed. Any existing RSA keys are not exportable. New keys are generated as nonexportable by default. It is not possible to convert an existing nonexportable key to an exportable key.

As of Cisco IOS Release 12.2(15)T, users can share the private RSA key pair of a router with standby routers, therefore transferring the security credentials between networking devices. The key pair that is shared between two routers will allow one router to immediately and transparently take over the functionality of the other router. If the main router were to fail, the standby router could be dropped into the network to replace the failed router without the need to regenerate keys, reenroll with the CA, or manually redistribute keys.

Exporting and importing an RSA key pair also enables users to place the same RSA key pair on multiple routers so that all management stations using Secure Shell (SSH) can be configured with a single public RSA key.

Exportable RSA Keys in PEM-Formatted Files

Using privacy-enhanced mail (PEM)-formatted files to import or export RSA keys can be helpful for customers who are running Cisco IOS software Release 12.3(4)T or later and who are using secure socket layer (SSL) or secure shell (SSH) applications to manually generate RSA key pairs and import the keys back into their PKI applications. PEM-formatted files allow customers to directly use existing RSA key pairs on their Cisco IOS routers instead of generating new keys.

Passphrase Protection While Importing and Exporting RSA Keys

You have to include a passphrase to encrypt the PKCS12 file or the PEM file that will be exported, and when the PKCS12 or PEM file is imported, the same passphrase has to be entered to decrypt it. Encrypting the

PKCS12 or PEM file when it is being exported, deleted, or imported protects the file from unauthorized access and use while it is being transported or stored on an external device.

The passphrase can be any phrase that is at least eight characters in length; it can include spaces and punctuation, excluding the question mark (?), which has special meaning to the Cisco IOS parser.

How to Convert an Exportable RSA Key Pair to a Nonexportable RSA Key Pair

Passphrase protection protects the external PKCS12 or PEM file from unauthorized access and use. To prevent an RSA key pair from being exported, it must be labeled “nonexportable.” To convert an exportable RSA key pair into a nonexportable key pair, the key pair must be exported and then reimported without specifying the “exportable” keyword.

How to Set Up and Deploy RSA Keys Within a PKI

Generating an RSA Key Pair

Perform this task to manually generate an RSA key pair.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*]
4. **exit**
5. **show crypto key mypubkey rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*] **Example:**	(Optional) Generates the RSA key pair for the certificate server. • The **storage** keyword specifies the key storage location. • When specifying a label name by specifying the *key-label* argument, you must use the same name for

	Command or Action	Purpose
	`Router(config)#  crypto key generate rsa usage-keys modulus 2048`	the label that you plan to use for the certificate server (through the **crypto pki server** *cs-label*command). If a *key-label* argument is not specified, the default value, which is the fully qualified domain name (FQDN) of the router, is used. If the exportable RSA key pair is manually generated after the CA certificate has been generated, and before issuing the **no shutdown** command, then use the **crypto ca export pkcs12** command to export a PKCS12 file that contains the certificate server certificate and the private key. • By default, the modulus size of a CA key is 1024 bits. The recommended modulus for a CA key is 2048 bits. The range for a modulus size of a CA key is from 360 to 4096 bits. • The **on** keyword specifies that the RSA key pair is created on the specified device, including a Universal Serial Bus (USB) token, local disk, or NVRAM. The name of the device is followed by a colon (:). **Note** Keys created on a USB token must be 2048 bits or less. **Caution** Do not manually generate an rsa keypair under trustpoint. If we want to manually generate the keys, generate the key pairs as usage-keys and not as general-purpose keys.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 5	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Displays the RSA public keys of your router. This step allows you to verify that the RSA key pair has been successfully generated.

What to Do Next

After you have successfully generated an RSA key pair, you can proceed to any of the additional tasks in this module to generate additional RSA key pairs, perform export and import of RSA key pairs, or configure additional security parameters for the RSA key pair (such as encrypting or locking the private key).

Managing RSA Key Pairs and Trustpoint Certificates

Perform this task to configure the router to generate and store multiple RSA key pairs, associate the key pairs with a trustpoint, and get the certificates for the router from the trustpoint.

Before you begin

You must have already generated an RSA key pair as shown in the task "Generating an RSA Key Pair task."

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **rsakeypair** *key-label* [*key-size* [*encryption-key-size*]]
5. **enrollment selfsigned**
6. **subject-alt-name** *name*
7. **exit**
8. **cypto pki enroll** *name*
9. **exit**
10. **show crypto key mypubkey rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint TESTCA`	Creates a trustpoint and enters ca-trustpoint configuration mode.
Step 4	**rsakeypair** *key-label* [*key-size* [*encryption-key-size*]] **Example:** `Router(ca-trustpoint)# rsakeypair fancy-keys`	(Optional) The *key-label* argument specifies the name of the RSA key pair generated during enrollment (if it does not already exist or if the **auto-enroll regenerate** command is configured) to be used with the trustpoint certificate. By default, the fully qualified domain name (FQDN) key is used. • The keypair name cannot start from zero ('0'). For more details, see "How RSA Key Pairs are Associated with a Trustpoint" section. • (Optional) The *key-size* argument specifies the size of the RSA key pair. The recommended key size is 2048 bits.

	Command or Action	Purpose
		• (Optional) The *encryption-key-size* argument specifies the size of the second key, which is used to request separate encryption, signature keys, and certificates.
Step 5	**enrollment selfsigned** **Example:** `Router(ca-trustpoint)# enrollment selfsigned`	(Optional) Specifies self-signed enrollment for a trustpoint.
Step 6	**subject-alt-name** *name* **Example:** `Router(ca-trustpoint)# subject-alt-name TESTCA`	(Optional) The *name* argument specifies the trustpoint's name in the Subject Alternative Name (subjectAltName) field in the X.509 certificate, which is contained in the trustpoint certificate. By default, the Subject Alternative Name field is not included in the certificate. **Note** This X.509 certificate field is defined in RFC 2511. This option is used to create a self-signed trustpoint certificate for the router that contains the trustpoint name in the Subject Alternative Name (subjectAltName) field. This Subject Alternative Name can be used only when the **enrollment selfsigned** command is specified for self-signed enrollment in the trustpoint policy.

Step 7	**exit**
	Example:
	`Router`
	`(ca-trustpoint)#`
	`exit`
		Exits ca-trustpoint configuration mode.

Step 8	**cypto pki enroll** *name*
	Example:
	`Router(config)# cypto pki enroll`
	`TESTCA`
	Example:
	`% Include the router serial number in the subject`
	` name? [yes/no]: no`
	Example:
	`% Include an IP address in the subject name? [no]:`
	Example:
	`Generate Self Signed Router Certificate? [yes/no]:`
	` yes`
		Requests the certificates for the router from the trustpoint.
		The *name* argument specifies the trustpoint name. Once this command is entered, answer the prompts.
		Note Use the same trustpoint name entered with the **crypto pki trustpoint**command.

	Command or Action	Purpose
	Example: `Router Self Signed Certificate successfully created`	
Step 9	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 10	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Displays the RSA public keys of your router. This step allows you to verify that the RSA key pair has been successfully generated.

Example

The following example shows how to create a self-signed trustpoint certificate for the router that contains the trustpoint name in the Subject Alternative Name (subjectAltName) field:

```
Router> enable
Router# configure terminal
Router(config)#crypto pki trustpoint TESTCA
Router(ca-trustpoint)#hash sha256
Router(ca-trustpoint)#rsakeypair testca-rsa-key 2048
Router(ca-trustpoint)#exit
Router(config)#crypto pki enroll TESTCA
% Include the router serial number in the subject name? [yes/no]:no
% Include an IP address in the subject name? [no]: no
Generate Self Signed Router Certificate? [yes/no]: yes

Router Self Signed Certificate successfully created

Router(config)#
Router(config)#exit
Router#
```

The following certificate is created:

```
Router#show crypto pki certificate verbose Router Self-Signed Certificate
   Status: Available
   Version: 3
   Certificate Serial Number (hex): 01
   Certificate Usage: General Purpose
   Issuer:
     hostname=Router.cisco.com
   Subject:
     Name: Router.cisco.com
     hostname=Router.cisco.com
   Validity Date:
     start date: 11:41:50 EST Aug 13 2012
     end   date: 19:00:00 EST Dec 31 2019
   Subject Key Info:
     Public Key Algorithm: rsaEncryption
     RSA Public Key: (2048 bit)
   Signature Algorithm: SHA256 with RSA Encryption
   Fingerprint MD5: CA92D937 593BF19A 5B7F8466 F554D631
   Fingerprint SHA1: 57A9D411 2DDFAC81 68260F2F C6C8D7CF 4833F3E9
   X509v3 extensions:
     X509v3 Subject Key ID: 44340F76 A6B8DC37 80724650 0672875F 741D518C
     X509v3 Basic Constraints:
         CA: TRUE
     X509v3 Authority Key ID: 44340F76 A6B8DC37 80724650 0672875F 741D518C
     Authority Info Access:
   Associated Trustpoints: TESTCA

-----BEGIN CERTIFICATE-----
MIIBszCCAV2gAwIBAgIBAjANBgkqhkiG9w0BAQQFADAuMQ8wDQYDVQQDEwZURVNU
Q0ExGzAZBgkqhkiG9w0BCQIWDHIxLmNpc2NvLmNvbTAeFw0xMDAzMjIyMDI2MjBa
Fw0yMDAxMDEwMDAwMDBaMC4xDzANBgNVBAMTBlRFU1RDQTEbMBkGCSqGSIb3DQEJ
AhYMcjEuY2lzY28uY29tMFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBAI1xLjvrouLz
RNm8qYWI9Km9yX/wafXndY8A8o4+L8pexQhDlYyiaq7OoK6CYWH/ToyPidFW2DU0
t5WTGnIDcfsCAwEAAaNmMGQwDwYDVR0TAQH/BAUwAwEB/zARBgNVHREECjAIggZU
RVNUQ0EwHwYDVR0jBBgwFoAU+aSVh1+kyn1l+r44IFUY+Uxs1fMwHQYDVR0OBBYE
FPmklYdfpMp9Zfq+OCBVGPlMbNXzMA0GCSqGSIb3DQEBBAUAA0EAbZLnqKUaWu8T
WAIbeReTQTfJLZ8ao/U6cwXN0QKEQ37ghAdGVflFWVG6JUhv2OENNUQHXBYXNUWZ
4oBuU+U1dg==
-----END CERTIFICATE-----
```

Exporting and Importing RSA Keys

This section contains the following tasks that can be used for exporting and importing RSA keys. Whether you are using PKCS12 files or PEM files, exportable RSA keys allow you to use existing RSA keys on Cisco IOS routers instead of having to generate new RSA keys if the main router were to fail.

Exporting and Importing RSA Keys in PKCS12 Files

Exporting and importing RSA key pairs enables users to transfer security credentials between devices. The key pair that is shared between two devices allows one device to immediately and transparently take over the functionality of the other router.

Before you begin

You must generate an RSA key pair and mark it "exportable" as specified in the "Generating an RSA Key Pair" task.

Note

- You cannot export RSA keys that existed on the router before your system was upgraded to Cisco IOS Release 12.2(15)T or later. You have to generate new RSA keys and label them as "exportable" after you upgrade the Cisco IOS software.
- When you import a PKCS12 file that was generated by a third-party application, the PKCS12 file must include a CA certificate.
- If you want reexport an RSA key pair after you have already exported the key pair and imported them to a target router, you must specify the **exportable** keyword when you are importing the RSA key pair.
- The largest RSA key a router may import is 2048-bits.

SUMMARY STEPS

1. **crypto pki trustpoint** *name*
2. **rsakeypair** *key-label* [*key-size* [*encryption-key-size*]]
3. **exit**
4. **crypto pki export** *trustpointname* **pkcs12** *destination-url* **password** *password-phrase*
5. **crypto pki import** *trustpointname* **pkcs12** *source-url* **password** *password-phrase*
6. **exit**
7. **show crypto key mypubkey rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint my-ca`	Creates the trustpoint name that is to be associated with the RSA key pair and enters ca-trustpoint configuration mode.
Step 2	**rsakeypair** *key-label* [*key-size* [*encryption-key-size*]] **Example:** `Router(ca-trustpoint)# rsakeypair my-keys`	Specifies the key pair that is to be used with the trustpoint.
Step 3	**exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode.
Step 4	**crypto pki export** *trustpointname* **pkcs12** *destination-url* **password** *password-phrase* **Example:** `Router(config)# crypto pki export my-ca pkcs12 tftp://tftpserver/my-keys password mypassword123`	Exports the RSA keys through the trustpoint name. • The *trustpointname* argument enters the name of the trustpoint that issues the certificate that a user is going to export. When exporting the PKCS12 file, the trustpoint name is the RSA key name. • The *destination-url* argument enters the file system location of the PKCS12 file to which a user wants to import the RSA key pair. • The *password -phrase* argument must be entered to encrypt the PKCS12 file for export.
Step 5	**crypto pki import** *trustpointname* **pkcs12** *source-url* **password** *password-phrase* **Example:** `Router(config)# crypto pki import my-ca pkcs12 tftp://tftpserver/my-keys password mypassword123`	Imports the RSA keys to the target router. • The *trustpointname* argument enters the name of the trustpoint that issues the certificate that a user is going to export or import. When importing, the trustpoint becomes the RSA key name.

	Command or Action	Purpose
		• The *source-url* argument specifies the file system location of the PKCS12 file to which a user wants to export the RSA key pair. • The *password -phrase* must be entered to undo encryption when the RSA keys are imported.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Displays the RSA public keys of your router.

Exporting and Importing RSA Keys in PEM-Formatted Files

Perform this task to export or import RSA key pairs in PEM files.

Before you begin

You must generate an RSA key pair and mark it "exportable" as specified the "Generating an RSA Key Pair" task.

Note

- You cannot export and import RSA keys that were generated without an exportable flag before your system was upgraded to Cisco IOS Release 12.3(4)T or a later release. You have to generate new RSA keys after you upgrade the Cisco IOS software.
- The largest RSA key a router may import is 2048 bits.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

SUMMARY STEPS

1. **crypto key generate rsa** {**usage-keys** | **general-keys**} **label** *key-label* [**exportable**]
2. **crypto pki export** *trustpoint* **pem** {**terminal** | **url** *destination-url*} {**3des** | **des**} **password** *password-phrase*
3. **crypto pki import** *trustpoint* **pem** [**check** | **exportable** | *usage-keys*] {**terminal** | **url** *source-url*} **password***password-phrase*
4. **exit**
5. **show crypto key mypubkey rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**crypto key generate rsa** {**usage-keys** | **general-keys**} **label** *key-label* [**exportable**] **Example:** `Router(config)# crypto key generate rsa general-keys label mykey exportable`	Generates the RSA key pair. To use PEM files, the RSA key pair must be labeled exportable.
Step 2	**crypto pki export** *trustpoint* **pem** {**terminal** | **url** *destination-url*} {**3des** | **des**} **password** *password-phrase* **Example:** `Router(config)# crypto pki export mycs pem url nvram: 3des password mypassword123`	Exports the certificates and RSA keys that are associated with a trustpoint in a PEM-formatted file. • Enter the *trustpoint* name that is associated with the exported certificate and RSA key pair. The trustpoint name must match the name that was specified through the **crypto pki trustpoint** command • Use the **terminal** keyword to specify the certificate and RSA key pair that is displayed in PEM format on the console terminal. • Use the **url** keyword and *destination -url* argument to specify the URL of the file system where your router should export the certificates and RSA key pair. • (Optional) the **3des** keyword exports the trustpoint using the Triple Data Encryption Standard (3DES) encryption algorithm. • (Optional) the **des** keyword exports the trustpoint using the DES encryption algorithm. • Use the *password-phrase* argument to specify the encrypted password phrase that is used to encrypt the PEM file for import. **Tip** Be sure to keep the PEM file safe. For example, you may want to store it on another backup router.
Step 3	**crypto pki import** *trustpoint* **pem** [**check** | **exportable** | *usage-keys*] {**terminal** | **url** *source-url*} **password***password-phrase* **Example:** `Router(config)# crypto pki import mycs2 pem url nvram: password mypassword123`	Imports certificates and RSA keys to a trustpoint from PEM-formatted files. • Enter the *trustpoint* name that is associated with the imported certificate and RSA key pair. The trustpoint name must match the name that was specified through the **crypto pki trustpoint** command • (Optional) Use the **check** keyword to specify that an outdated certificate is not allowed.

	Command or Action	Purpose
		• (Optional) Use the **exportable** keyword to specify that the imported RSA key pair can be exported again to another Cisco device such as a router. • (Optional) Use the *usage-keys* argument to specify that two RSA special usage key pairs will be imported (that is, one encryption pair and one signature pair), instead of one general-purpose key pair. • Use the *source-url* argument to specify the URL of the file system where your router should import the certificates and RSA key pairs. • Use the *password-phrase* argument to specify the encrypted password phrase that is used to encrypt the PEM file for import. **Note** The password phrase can be any phrase that is at least eight characters in length; it can include spaces and punctuation, excluding the question mark (?), which has special meaning to the Cisco IOS parser. **Note** If you do not want the key to be exportable from your CA, import it back to the CA after it has been exported as a nonexportable key pair. Thus, the key cannot be taken off again.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 5	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Displays the RSA public keys of your router.

Encrypting and Locking Private Keys on a Router

Digital signatures are used to authenticate one device to another device. To use digital signatures, private information (the private key) must be stored on the device that is providing the signature. The stored private information may aid an attacker who steals the hardware device that contains the private key; for example, a thief might be able to use the stolen router to initiate a secure connection to another site by using the RSA private keys stored in the router.

Note RSA keys are lost during password recovery operations. If you lose your password, the RSA keys will be deleted when you perform the password recovery operation. (This function prevents an attacker from performing password recovery and then using the keys.)

To protect the private RSA key from an attacker, a user can encrypt the private key that is stored in NVRAM via a passphrase. Users can also "lock" the private key, which blocks new connection attempts from a running router and protects the key in the router if the router is stolen by an attempted attacker.

Perform this task to encrypt and lock the private key that is saved to NVRAM.

Note The RSA keys must be unlocked while enrolling the CA. The keys can be locked while authenticating the router with the CA because the private key of the router is not used during authentication.

Before you begin

Before encrypting or locking a private key, you should perform the following tasks:

- Generate an RSA key pair as shown in Generating an RSA Key Pair section.
- Optionally, you can authenticate and enroll each router with the CA server.

Note **Backward Compatibility Restriction**

Any image prior to Cisco IOS Release 12.3(7)T does not support encrypted keys. To prevent your router from losing all encrypted keys, ensure that only unencrypted keys are written to NVRAM before booting an image prior to Cisco IOS Release 12.3(7)T.

If you must download an image prior to Cisco IOS Release 12.3(7)T, decrypt the key and immediately save the configuration so the downloaded image does not overwrite the configuration.

Interaction with Applications

An encrypted key is not effective after the router boots up until you manually unlock the key (via the **crypto key unlock rsa** command). Depending on which key pairs are encrypted, this functionality may adversely affect applications such as IP security (IPsec), SSH, and SSL; that is, management of the router over a secure channel may not be possible until the necessary key pair is unlocked.

>

SUMMARY STEPS

1. crypto key encrypt [write] rsa [name key-name] passphrase passphrase
2. **exit**
3. **show crypto key mypubkey rsa**
4. **crypto key lock rsa name** *key-name*] **passphrase** *passphrase*
5. **show crypto key mypubkey rsa**
6. **crypto key unlock rsa** [**name** *key-name*] **passphrase** *passphrase*

7. **configure terminal**
8. **crypto key decrypt** [**write**] **rsa** [**name***key-name*] **passphrase** *passphrase*

DETAILED STEPS

	Command or Action	Purpose
Step 1	crypto key encrypt [write] rsa [name key-name] passphrase passphrase **Example:** `Router(config)# crypto key encrypt write rsa name pki.example.com passphrase password`	Encrypts the RSA keys. After this command is issued, the router can continue to use the key; the key remains unlocked. **Note** If the **write** keyword is not issued, the configuration must be manually written to NVRAM; otherwise, the encrypted key will be lost next time the router is reloaded.
Step 2	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 3	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Shows that the private key is encrypted (protected) and unlocked. **Note** You can also use this command to verify that applications such as Internet Key Exchange (IKE) and SSH are properly working after the key has been encrypted.
Step 4	**crypto key lock rsa name** *key-name*] **passphrase** *passphrase* **Example:** `Router# crypto key lock rsa name pki.example.com passphrase password`	(Optional) Locks the encrypted private key on a running router. **Note** After the key is locked, it cannot be used to authenticate the router to a peer device. This behavior disables any IPSec or SSL connections that use the locked key. Any existing IPSec tunnels created on the basis of the locked key will be closed. If all RSA keys are locked, SSH will automatically be disabled.
Step 5	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Shows that the private key is protected and locked. The output will also show failed connection attempts via applications such as IKE, SSH, and SSL.
Step 6	**crypto key unlock rsa** [**name** *key-name*] **passphrase** *passphrase* **Example:** `Router# crypto key unlock rsa name pki.example.com passphrase password`	(Optional) Unlocks the private key. **Note** After this command is issued, you can continue to establish IKE tunnels.

	Command or Action	Purpose
Step 7	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 8	**crypto key decrypt** [**write**] **rsa** [**name***key-name*] **passphrase** *passphrase* **Example:** `Router(config)# crypto key decrypt write rsa name pki.example.com passphrase password`	(Optional) Deletes the encrypted key and leaves only the unencrypted key. **Note** The **write** keyword immediately saves the unencrypted key to NVRAM. If the **write** keyword is not issued, the configuration must be manually written to NVRAM; otherwise, the key will remain encrypted the next time the router is reloaded.

Removing RSA Key Pair Settings

An RSA key pair may need to be removed for one of the following reasons:

- During manual PKI operations and maintenance, old RSA keys can be removed and replaced with new keys.
- An existing CA is replaced and the new CA requires newly generated keys; for example, the required key size might have changed in an organization so you would have to delete the old 1024-bit keys and generate new 2048-bit keys.
- **T** he peer router's public keys can be deleted in order to help debug signature verification problems in IKEv1 and IKEv2. Keys are cached by default with the lifetime of the certificate revocation list (CRL) associated with the trustpoint.

Perform this task to remove all RSA keys or the specified RSA key pair that has been generated by your router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. crypto key zeroize rsa [key-pair-label]
4. **crypto key zeroize pubkey-chain** [*index*]
5. **exit**
6. **show crypto key mypubkey rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	crypto key zeroize rsa [key-pair-label] **Example:** `Router(config)# crypto key zeroize rsa fancy-keys`	Deletes RSA key pairs from your router. • If the *key-pair-label* argument is not specified, all RSA keys that have been generated by your router will be deleted.
Step 4	**crypto key zeroize pubkey-chain** [*index*] **Example:** `Router(config)# crypto key zeroize pubkey-chain`	Deletes the remote peer's public key from the cache. (Optional) Use the *index* argument to delete a particular public key index entry. If no index entry is specified, then all the entries are deleted. The acceptable range of index entries is from 1 to 65535.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 6	**show crypto key mypubkey rsa** **Example:** `Router# show crypto key mypubkey rsa`	(Optional) Displays the RSA public keys of your router. This step allows you to verify that the RSA key pair has been successfully generated.

Configuration Examples for RSA Key Pair Deployment

Generating and Specifying RSA Keys Example

The following example is a sample trustpoint configuration that shows how to generate and specify the RSA key pair "exampleCAkeys":

```
crypto key generate rsa general-purpose exampleCAkeys
crypto ca trustpoint exampleCAkeys
 enroll url http://exampleCAkeys/certsrv/mscep/mscep.dll
 rsakeypair exampleCAkeys 1024 1024
```

Exporting and Importing RSA Keys Examples

Exporting and Importing RSA Keys in PKCS12 Files Example

In the following example, an RSA key pair “mynewkp” is generated on Router A, and a trustpoint name “mynewtp” is created and associated with the RSA key pair. The trustpoint is exported to a TFTP server, so that it can be imported on Router B. By importing the trustpoint “mynewtp” to Router B, the user has imported the RSA key pair “mynewkp” to Router B.

Router A

```
crypto key generate rsa general label mykeys exportable
! The name for the keys will be:mynewkp
Choose the size of the key modulus in the range of 360 to 2048 for your
General Purpose Keys. Choosing a key modulus greater than 512 may take
a few minutes.
How many bits in the modulus [512]: 2048
% Generating 2048 bit RSA keys ...[OK]
!
crypto pki trustpoint mynewtp
 rsakeypair mykeys
 exit
crypto pki export mytp pkcs12 flash:myexport password mypassword123
Destination filename [myexport]?
Writing pkcs12 file to tftp:/mytftpserver/myexport
CRYPTO_PKI:Exported PKCS12 file successfully.
Verifying checksum... OK (0x3307)
!
July 8 17:30:09 GMT:%CRYPTO-6-PKCS12EXPORT_SUCCESS:PKCS #12 Successfully Exported.
```

Router B

```
crypto pki import mynewtp pkcs12 flash:myexport password mypassword123
Source filename [myexport]?
CRYPTO_PKI:Imported PKCS12 file successfully.
!
July 8 18:07:50 GMT:%CRYPTO-6-PKCS12IMPORT_SUCCESS:PKCS #12 Successfully Imported.
```

Exporting and Importing and RSA Keys in PEM Files Example

The following example shows the generation, exportation, and importation fo the RSA key pair "mytp", and verifies its status:

```
! Generate the key pair
!
Router(config)# crypto key generate rsa general-purpose label mytp exportable

The name for the keys will be: mytp
Choose the size of the key modulus in the range of 360 to 2048 for your
General Purpose Keys. Choosing a key modulus greater than 512 may take a few minutes.
How many bits in the modulus [512]: 2048
% Generating 2048 bit RSA keys ...[OK]
!
! Archive the key pair to a remote location, and use a good password.
!
Router(config)# crypto pki export mytp pem url nvram:mytp 3des password mypassword123
```

```
% Key name:mytp
Usage:General Purpose Key
Exporting public key...
Destination filename [mytp.pub]?
Writing file to nvram:mytp.pub
Exporting private key...
Destination filename [mytp.prv]?
Writing file to nvram:mytp.prv
!
! Import the key as a different name.
!
Router(config)# crypto pki import mytp2 pem url nvram:mytp2 password mypassword123

% Importing public key or certificate PEM file...
Source filename [mytp2.pub]?
Reading file from nvram:mytp2.pub
% Importing private key PEM file...
Source filename [mytp2.prv]?
Reading file from nvram:mytp2.prv% Key pair import succeeded.
!
! After the key has been imported, it is no longer exportable.
!
! Verify the status of the key.
!
Router# show crypto key mypubkey rsa

% Key pair was generated at:18:04:56 GMT Jun 6 2011
Key name:mycs
Usage:General Purpose Key
Key is exportable.
Key Data:
30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00E65253
9C30C12E 295AB73F B1DF9FAD 86F88192 7D4FA4D2 8BA7FB49 9045BAB9 373A31CB
A6B1B8F4 329F2E7E 8A50997E AADBCFAA 23C29E19 C45F4F05 DBB2FA51 4B7E9F79
A1095115 759D6BC3 5DFB5D7F BCF655BF 6317DB12 A8287795 7D8DC6A3 D31B2486
C9C96D2C 2F70B50D 3B4CDDAE F661041A 445AE11D 002EEF08 F2A627A0 5B020301 0001
% Key pair was generated at:18:17:25 GMT Jun 6 2011
Key name:mycs2
Usage:General Purpose Key
Key is not exportable.
Key Data:
30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00E65253
9C30C12E 295AB73F B1DF9FAD 86F88192 7D4FA4D2 8BA7FB49 9045BAB9 373A31CB
A6B1B8F4 329F2E7E 8A50997E AADBCFAA 23C29E19 C45F4F05 DBB2FA51 4B7E9F79
A1095115 759D6BC3 5DFB5D7F BCF655BF 6317DB12 A8287795 7D8DC6A3 D31B2486
C9C96D2C 2F70B50D 3B4CDDAE F661041A 445AE11D 002EEF08 F2A627A0 5B020301 0001
```

Exporting Router RSA Key Pairs and Certificates from PEM Files Example

The following example shows how to generate and export the RSA key pair “aaa” and certificates of the router in PEM files that are associated with the trustpoint “mycs.” This example also shows PEM-formatted files, which include PEM boundaries before and after the base64-encoded data, that are used by other SSL and SSH applications.

```
Router(config)# crypto key generate rsa general-keys label aaa exportable

The name for the keys will be:aaa
Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose
Keys. Choosing a key modulus greater than 512 may take a few minutes.
!
How many bits in the modulus [512]:
% Generating 512 bit RSA keys ...[OK]
```

```
!
Router(config)# crypto pki trustpoint mycs

Router(ca-trustpoint)# enrollment url http://mycs

Router(ca-trustpoint)#
rsakeypair aaa

Router(ca-trustpoint)# exit

Router(config)# crypto pki authenticate mycs

Certificate has the following attributes:
Fingerprint:C21514AC 12815946 09F635ED FBB6CF31
% Do you accept this certificate? [yes/no]: y
Trustpoint CA certificate accepted.
!
Router(config)# crypto pki enroll mycs

%
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this password to the CA
Administrator in order to revoke your certificate.
For security reasons your password will not be saved in the configuration.
Please make a note of it.
Password:
Re-enter password:
% The fully-qualified domain name in the certificate will be: Router
% The subject name in the certificate will be:host.example.com
% Include the router serial number in the subject name? [yes/no]: n
% Include an IP address in the subject name? [no]: n
Request certificate from CA? [yes/no]: y
% Certificate request sent to Certificate Authority
% The certificate request fingerprint will be displayed.
% The 'show crypto ca certificate' command will also show the fingerprint.
Router(config)# Fingerprint:8DA777BC 08477073 A5BE2403 812DD157
00:29:11:%CRYPTO-6-CERTRET:Certificate received from Certificate Authority
Router(config)# crypto ca export aaa pem terminal 3des password

% CA certificate:
-----BEGIN CERTIFICATE-----
MIICAzCCAa2gAwIBAgIBATANBgkqhkiG9w0BAQUFADBOMQswCQYDVQQGEwJVUzES
<snip>
waDeNOSI3WlDa0AWq5DkVBkxwgn0TqIJXJOCttjHnWHK1LMcMVGn
-----END CERTIFICATE-----
% Key name:aaa
Usage:General Purpose Key
-----BEGIN RSA PRIVATE KEY-----
Proc-Type:4,ENCRYPTED
DEK-Info:DES-EDE3-CBC,ED6B210B626BC81A
Urguv0jnjwOgowWVUQ2XR5nbzzYHI2vGLunpH/IxIsJuNjRVjbAAUpGk7VnPCT87
<snip>
kLCOtxzEv7JHc72gMku9uUlrLSnFH5slzAtoC0czfU4=
-----END RSA PRIVATE KEY-----
% Certificate:
-----BEGIN CERTIFICATE-----
MIICTjCCAfigAwIBAgICIQUwDQYJKoZIhvcNAQEFBQAwTjELMAkGA1UEBhMCVVMx
<snip>
6xlBaIsuMxnHmr89KkKkYlU6
-----END CERTIFICATE-----
```

Importing Router RSA Key Pairs and Certificate from PEM Files Example

The following example shows how to import the RSA key pairs and certificate to the trustpoint "ggg" from PEM files via TFTP:

```
Router(config)# crypto pki import ggg pem url tftp://10.1.1.2/username/msca password

% Importing CA certificate...
Address or name of remote host [10.1.1.2]?
Destination filename [username/msca.ca]?
Reading file from tftp://10.1.1.2/username/msca.ca
Loading username/msca.ca from 10.1.1.2 (via Ethernet0):!
[OK - 1082 bytes]
% Importing private key PEM file...
Address or name of remote host [10.1.1.2]?
Destination filename [username/msca.prv]?
Reading file from tftp://10.1.1.2/username/msca.prv
Loading username/msca.prv from 10.1.1.2 (via Ethernet0):!
[OK - 573 bytes]
% Importing certificate PEM file...
Address or name of remote host [10.1.1.2]?
Destination filename [username/msca.crt]?
Reading file from tftp://10.1.1.2/username/msca.crt
Loading username/msca.crt from 10.1.1.2 (via Ethernet0):!
[OK - 1289 bytes]
% PEM files import succeeded.
Router(config)#
```

Encrypting and Locking Private Keys on a Router Examples

Configuring and Verifying an Encrypted Key Example

The following example shows how to encrypt the RSA key "pki-123.example.com." Thereafter, the **show crypto key mypubkey rsa** command is issued to verify that the RSA key is encrypted (protected) and unlocked.

```
Router(config)# crypto key encrypt rsa name pki-123.example.com passphrase password
Router(config)# exit
Router# show crypto key mypubkey rsa
```

% Key pair was generated at:00:15:32 GMT Jun 25 2003

Key name:pki-123.example.com

Usage:General Purpose Key

*** The key is protected and UNLOCKED. ***

Key is not exportable.

Key Data:

305C300D 06092A86 4886F70D 01010105 00034B00 30480241 00E0CC9A 1D23B52C

CD00910C ABD392AE BA6D0E3F FC47A0EF 8AFEE340 0EC1E62B D40E7DCC

23C4D09E

03018B98 E0C07B42 3CFD1A32 2A3A13C0 1FF919C5 8DE9565F 1F020301 0001

% Key pair was generated at:00:15:33 GMT Jun 25 2003

```
Key name:pki-123.example.com.server

Usage:Encryption Key

Key is exportable.

Key Data:

307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00D3491E 2A21D383

854D7DA8 58AFBDAC 4E11A7DD E6C40AC6 66473A9F 0C845120 7C0C6EC8 1FFF5757

3A41CE04 FDCB40A4 B9C68B4F BC7D624B 470339A3 DE739D3E F7DDB549 91CD4DA4

DF190D26 7033958C 8A61787B D40D28B8 29BCD0ED 4E6275C0 6D020301 0001

Router#
```

Configuring and Verifying a Locked Key Example

The following example shows how to lock the key “pki-123.example.com.” Thereafter, the **show crypto key mypubkey rsa** command is issued to verify that the key is protected (encrypted) and locked.

```
Router# crypto key lock rsa name pki-123.example.com passphrase password
!
Router# show crypto key mypubkey rsa

% Key pair was generated at:20:29:41 GMT Jun 20 2003
Key name:pki-123.example.com
Usage:General Purpose Key
*** The key is protected and LOCKED. ***
Key is exportable.
Key Data:
305C300D 06092A86 4886F70D 01010105 00034B00 30480241 00D7808D C5FF14AC
0D2B55AC 5D199F2F 7CB4B355 C555E07B 6D0DECBE 4519B1F0 75B12D6F 902D6E9F
B6FDAD8D 654EF851 5701D5D7 EDA047ED 9A2A619D 5639DF18 EB020301 0001
```

Where to Go Next

After you have generated an RSA key pair, you should set up the trustpoint. If you have already set up the trustpoint, you should authenticate and enroll the routers in a PKI. For information on enrollment, see the module “Configuring Certificate Enrollment for a PKI.”

Additional References

Related Documents

Related Topic	Document Title
Overview of PKI, including RSA keys, certificate enrollment, and CAs	Cisco IOS PKI Overview: Understanding and Planning a PKI
PKI commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*

Related Topic	Document Title
Recommended cryptographic algorithms	*Next Generation Encryption*

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2409	*The Internet Key Exchange (IKE)*
RFC 2511	Internet X.509 Certificate Request Message Format

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSA Keys Within a PKI

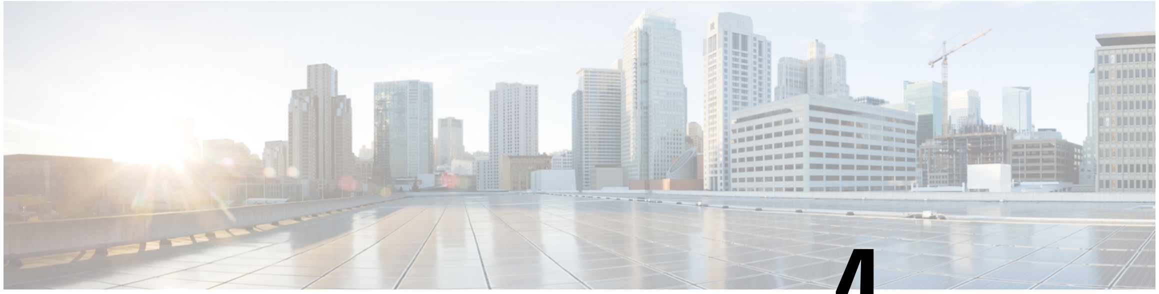
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for RSA Keys Within a PKI

Feature Name	Software Releases	Feature Configuration Information
Cisco IOS 4096-Bit Public Key Support		This feature introduces Cisco IOS 4096-bit peer public key support.

Feature Name	Software Releases	Feature Configuration Information
Exporting and Importing RSA Keys		This feature allows you to transfer security credentials between devices by exporting and importing RSA keys. The key pair that is shared between two devices will allow one device to immediately and transparently take over the functionality of the other router. The following commands were introduced or modified by this feature: **crypto ca export pkcs12**, **crypto ca import pkcs12**, **crypto key generate rsa (IKE)**
Import of RSA Key Pair and Certificates in PEM Format		This feature allows customers to use PEM-formatted files to import or export RSA key pairs. PEM-formatted files allow customers to directly use existing RSA key pairs on their Cisco IOS routers instead of generating new keys. The following commands were introduced by this feature: **crypto ca export pem**, **crypto ca import pem**, **crypto key export pem**, **crypto key import pem**
Multiple RSA Key Pair Support		This feature allows a user to configure a router to have multiple RSA key pairs. Thus, the Cisco IOS software can maintain a different key pair for each identity certificate. The following commands were introduced or modified by this feature: **crypto key generate rsa**, **crypto key zeroize rsa**, **rsakeypair**
Protected Private Key Storage		This feature allows a user to encrypt and lock the RSA private keys that are used on a Cisco IOS router, thereby, preventing unauthorized use of the private keys. The following commands were introduced or modified by this feature : **crypto key decrypt rsa**, **crypto key encrypt rsa**, **crypto key lock rsa**, **crypto key unlock rsa**, **show crypto key mypubkey rsa**



CHAPTER 4

Configuring Authorization and Revocation of Certificates in a PKI

This module describes how to configure authorization and revocation of certificates in a public key infrastructure (PKI). It includes information on high-availability support for the certificate server.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Authorization and Revocation of Certificates

Plan Your PKI Strategy

Tip It is strongly recommended that you plan your entire PKI strategy before you begin to deploy actual certificates.

Authorization and revocation can occur only after you or a network administrator have completed the following tasks:

- Configured the certificate authority (CA).
- Enrolled peer devices with the CA.
- Identified and configured the protocol (such as IP Security [IPsec] or secure socket layer [SSL]) that is to be used for peer-to-peer communication.

You should decide which authorization and revocation strategy you are going to configure before enrolling peer devices because the peer device certificates might have to contain authorization and revocation-specific information.

“crypto ca” to “crypto pki” CLI Change

As of Cisco IOS Release 12.3(7)T, all commands that begin as “crypto ca” have been changed to begin as “crypto pki.” Although the router will still accept crypto ca commands, all output will be read back as crypto pki.

High Availability

For high availability, IPsec-secured Stream Control Transmission Protocol (SCTP) must be configured on both the active and the standby routers. For synchronization to work, the redundancy mode on the certificate servers must be set to ACTIVE/STANDBY after you configure SCTP.

Restrictions for Authorization and Revocation of Certificates

- PKI High Availability (HA) support of intra-chassis stateful switchover (SSO) redundancy is currently not supported on all switches running the Cisco IOS Release 12.2 S software. See Cisco bug CSCtb59872 for more information.
- Depending on your Cisco IOS release, Lightweight Directory Access Protocol (LDAP) is supported.

Information About Authorization and Revocation of Certificates

PKI Authorization

PKI authentication does not provide authorization. Current solutions for authorization are specific to the router that is being configured, although a centrally managed solution is often required.

There is not a standard mechanism by which certificates are defined as authorized for some tasks and not for others. This authorization information can be captured in the certificate itself if the application is aware of the certificate-based authorization information. But this solution does not provide a simple mechanism for real-time updates to the authorization information and forces each application to be aware of the specific authorization information embedded in the certificate.

When the certificate-based ACL mechanism is configured as part of the trustpoint authentication, the application is no longer responsible for determining this authorization information, and it is no longer possible to specify for which application the certificate is authorized. In some cases, the certificate-based ACL on the router gets so large that it cannot be managed. Additionally, it is beneficial to retrieve certificate-based ACL indications from an external server.

Current solutions to the real-time authorization problem involve specifying a new protocol and building a new server (with associated tasks, such as management and data distribution).

PKI and AAA Server Integration for Certificate Status

Integrating your PKI with an authentication, authorization, and accounting (AAA) server provides an alternative online certificate status solution that leverages the existing AAA infrastructure. Certificates can be listed in the AAA database with appropriate levels of authorization. For components that do not explicitly support PKI-AAA, a default label of “all” from the AAA server provides authorization. Likewise, a label of “none” from the AAA database indicates that the specified certificate is not valid. (The absence of any application label is equivalent, but “none” is included for completeness and clarity). If the application component does support PKI-AAA, the component may be specified directly; for example, the application component could be “ipsec,” “ssl,” or “osp.” (ipsec=IP Security, ssl=Secure Sockets Layer, and osp=Open Settlement Protocol.)

Note Currently, no application component supports specification of the application label.

- There may be a time delay when accessing the AAA server. If the AAA server is not available, the authorization fails.

RADIUS or TACACS+ Choosing a AAA Server Protocol

The AAA server can be configured to work with either the RADIUS or TACACS+ protocol. When you are configuring the AAA server for the PKI integration, you must set the RADIUS or TACACS attributes that are required for authorization.

If the RADIUS protocol is used, the password that is configured for the username in the AAA server should be set to “cisco,” which is acceptable because the certificate validation provides authentication and the AAA database is only being used for authorization. When the TACACS protocol is used, the password that is configured for the username in the AAA server is irrelevant because TACACS supports authorization without requiring authentication (the password is used for authentication).

In addition, if you are using TACACS, you must add a PKI service to the AAA server. The custom attribute “cert-application=all” is added under the PKI service for the particular user or usergroup to authorize the specific username.

Attribute-Value Pairs for PKI and AAA Server Integration

The table below lists the attribute-value (AV) pairs that are to be used when setting up PKI integration with a AAA server. (Note the values shown in the table are possible values.) The AV pairs must match the client configuration. If they do not match, the peer certificate is not authorized.

Note Users can sometimes have AV pairs that are different from those of every other user. As a result, a unique username is required for each user. The **all** parameter (within the **authorization username** command) specifies that the entire subject name of the certificate will be used as the authorization username.

Table 2: AV Pairs That Must Match

AV Pair	Value
cisco-avpair=pki:cert-application=all	Valid values are “all” and “none.”
cisco-avpair=pki:cert-trustpoint=msca	The value is a Cisco IOS command-line interface (CLI) configuration trustpoint label. **Note** The cert-trustpoint AV pair is normally optional. If it is specified, the Cisco IOS router query must be coming from a certificate trustpoint that has a matching label, and the certificate that is authenticated must have the specified certificate serial number.
cisco-avpair=pki:cert-serial=16318DB7000100001671	The value is a certificate serial number. **Note** The cert-serial AV pair is normally optional. If it is specified, the Cisco IOS router query must be coming from a certificate trustpoint that has a matching label, and the certificate that is authenticated must have the specified certificate serial number.

AV Pair	Value
cisco-avpair=pki:cert-lifetime-end=1:00 jan 1, 2003	The cert-lifetime-end AV pair is available to artificially extend a certificate lifetime beyond the time period that is indicated in the certificate itself. If the cert-lifetime-end AV pair is used, the cert-trustpoint and cert-serial AV pairs must also be specified. The value must match the following form: hours:minutes month day, year. **Note** Only the first three characters of a month are used: Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec. If more than three characters are entered for the month, the remaining characters are ignored (for example Janxxxx).

CRLs or OCSP Server Choosing a Certificate Revocation Mechanism

After a certificate is validated as a properly signed certificate, a certificate revocation method is performed to ensure that the certificate has not been revoked by the issuing CA. Cisco IOS software supports two revocation mechanisms--certificate revocation lists (CRLs) and Online Certificate Status Protocol (OCSP). Cisco IOS software also supports AAA integration for certificate checking; however, additional authorization functionality is included. For more information on PKI and AAA certificate authorization and status check, see the PKI and AAA Server Integration for Certificate Status section.

The following sections explain how each revocation mechanism works:

What Is a CRL

A certificate revocation list (CRL) is a list of revoked certificates. The CRL is created and digitally signed by the CA that originally issued the certificates. The CRL contains dates for when each certificate was issued and when it expires.

CAs publish new CRLs periodically or when a certificate for which the CA is responsible has been revoked. By default, a new CRL is downloaded after the currently cached CRL expires. An administrator may also configure the duration for which CRLs are cached in router memory or disable CRL caching completely. The CRL caching configuration applies to all CRLs associated with a trustpoint.

When the CRL expires, the router deletes it from its cache. A new CRL is downloaded when a certificate is presented for verification; however, if a newer version of the CRL that lists the certificate under examination is on the server but the router is still using the CRL in its cache, the router does not know that the certificate has been revoked. The certificate passes the revocation check even though it should have been denied.

When a CA issues a certificate, the CA can include in the certificate the CRL distribution point (CDP) for that certificate. Cisco IOS client devices use CDPs to locate and load the correct CRL. The Cisco IOS client supports multiple CDPs, but the Cisco IOS CA currently supports only one CDP; however, third-party vendor CAs may support multiple CDPs or different CDPs per certificate. If a CDP is not specified in the certificate, the client device uses the default Simple Certificate Enrollment Protocol (SCEP) method to retrieve the CRL. (The CDP location can be specified through the **cdp-url**command.)

When implementing CRLs, you should consider the following design considerations:

- CRL lifetimes and the security association (SA) and Internet Key Exchange (IKE) lifetimes.
- The CRL lifetime determines the length of time between CA-issued updates to the CRL. The default CRL lifetime value, which is 168 hours [1 week], can be changed through the **lifetime crl** command.
- The method of the CDP determines how the CRL is retrieved; some possible choices include HTTP, Lightweight Directory Access Protocol (LDAP), SCEP, or TFTP. HTTP, TFTP, and LDAP are the most commonly used methods. Although Cisco IOS software defaults to SCEP, an HTTP CDP is recommended for large installations using CRLs because HTTP can be made highly scalable.
- The location of the CDP determines from where the CRL is retrieved; for example, you can specify the server and file path from which to retrieve the CRL.

Querying All CDPs During Revocation Check

When a CDP server does not respond to a request, the Cisco IOS software reports an error, which may result in the peer's certificate being rejected. To prevent a possible certificate rejection and if there are multiple CDPs in a certificate, the Cisco IOS software will attempt to use the CDPs in the order in which they appear in the certificate. The router will attempt to retrieve a CRL using each CDP URL or directory specification. If an error occurs using a CDP, an attempt will be made using the next CDP.

Note Prior to Cisco IOS Release 12.3(7)T, the Cisco IOS software makes only one attempt to retrieve the CRL, even when the certificate contains more than one CDP.

Tip Although the Cisco IOS software will make every attempt to obtain the CRL from one of the indicated CDPs, it is recommended that you use an HTTP CDP server with high-speed redundant HTTP servers to avoid application timeouts because of slow CDP responses.

What Is OCSP

OCSP is an online mechanism that is used to determine certificate validity and provides the following flexibility as a revocation mechanism:

- OCSP can provide real-time certificate status checking.
- OCSP allows the network administrator to specify a central OCSP server, which can service all devices within a network.
- OCSP also allows the network administrator the flexibility to specify multiple OCSP servers, either per client certificate or per group of client certificates.
- OCSP server validation is usually based on the root CA certificate or a valid subordinate CA certificate, but may also be configured so that external CA certificates or self-signed certificates may be used. Using external CA certificates or self-signed certificates allows the OCSP servers certificate to be issued and validated from an alternative PKI hierarchy.

A network administrator can configure an OCSP server to collect and update CRLs from different CA servers. The devices within the network can rely on the OCSP server to check the certificate status without retrieving and caching each CRL for every peer. When peers have to check the revocation status of a certificate, they send a query to the OCSP server that includes the serial number of the certificate in question and an optional unique identifier for the OCSP request, or a nonce. The OCSP server holds a copy of the CRL to determine if the CA has listed the certificate as being revoked; the server then responds to the peer including the nonce. If the nonce in the response from the OCSP server does not match the original nonce sent by the peer, the response is considered invalid and certificate verification fails. The dialog between the OCSP server and the peer consumes less bandwidth than most CRL downloads.

If the OCSP server is using a CRL, CRL time limitations will be applicable; that is, a CRL that is still valid might be used by the OCSP server although a new CRL has been issued by the CRL containing additional certificate revocation information. Because fewer devices are downloading the CRL information on a regular basis, you can decrease the CRL lifetime value or configure the OCSP server not to cache the CRL. For more information, check your OCSP server documentation.

When to Use an OCSP Server

OCSP may be more appropriate than CRLs if your PKI has any of the following characteristics:

- Real-time certificate revocation status is necessary. CRLs are updated only periodically and the latest CRL may not always be cached by the client device. For example, if a client does not yet have the latest CRL cached and a newly revoked certificate is being checked, that revoked certificate will successfully pass the revocation check.
- There are a large number of revoked certificates or multiple CRLs. Caching a large CRL consumes large portions of Cisco IOS memory and may reduce resources available to other processes.
- CRLs expire frequently, causing the CDP to handle a larger load of CRLs.

Note As of Cisco IOS Release 12.4(9)T or later, an administrator may configure CRL caching, either by disabling CRL caching completely or setting a maximum lifetime for a cached CRL per trustpoint.

When to Use Certificate-Based ACLs for Authorization or Revocation

Certificates contain several fields that are used to determine whether a device or user is authorized to perform a specified action.

Because certificate-based ACLs are configured on the device, they do not scale well for large numbers of ACLs; however, certificate-based ACLs do provide very granular control of specific device behavior. Certificate-based ACLs are also leveraged by additional features to help determine when PKI components such as revocation, authorization, or a trustpoint should be used. They provide a general mechanism allowing users to select a specific certificate or a group of certificates that are being validated for either authorization or additional processing.

Certificate-based ACLs specify one or more fields within the certificate and an acceptable value for each specified field. You can specify which fields within a certificate should be checked and which values those fields may or may not have.

There are six logical tests for comparing the field with the value--equal, not equal, contains, does not contain, less than, and greater than or equal. If more than one field is specified within a single certificate-based ACL, the tests of all of the fields within the ACL must succeed to match the ACL. The same field may be specified multiple times within the same ACL. More than one ACL may be specified, and ACL will be processed in turn until a match is found or all of the ACLs have been processed.

Ignore Revocation Checks Using a Certificate-Based ACL

Certificate-based ACLs can be configured to instruct your router to ignore the revocation check and expired certificates of a valid peer. Thus, a certificate that meets the specified criteria can be accepted regardless of the validity period of the certificate, or if the certificate meets the specified criteria, revocation checking does not have to be performed. You can also use a certificate-based ACL to ignore the revocation check when the communication with a AAA server is protected with a certificate.

Ignoring Revocation Lists

To allow a trustpoint to enforce CRLs except for specific certificates, enter the **match certificate**command with the **skip revocation-check** keyword. This type of enforcement is most useful in a hub-and-spoke configuration in which you also want to allow direct spoke-to-spoke connections. In pure hub-and-spoke configurations, all spokes connect only to the hub, so CRL checking is necessary only on the hub. For one spoke to communicate directly with another spoke, the **match certificate**command with the **skip revocation-check** keyword can be used for neighboring peer certificates instead of requiring a CRL on each spoke.

Ignoring Expired Certificates

To configure your router to ignore expired certificates, enter the **match certificate** command with the **allow expired-certificate** keyword. This command has the following purposes:

- If the certificate of a peer has expired, this command may be used to “allow” the expired certificate until the peer can obtain a new certificate.
- If your router clock has not yet been set to the correct time, the certificate of a peer will appear to be not yet valid until the clock is set. This command may be used to allow the certificate of the peer even though your router clock is not set.

Note If Network Time Protocol (NTP) is available only via the IPSec connection (usually via the hub in a hub-and-spoke configuration), the router clock can never be set. The tunnel to the hub cannot be “brought up” because the certificate of the hub is not yet valid.

- “Expired” is a generic term for a certificate that is expired or that is not yet valid. The certificate has a start and end time. An expired certificate, for purposes of the ACL, is one for which the current time of the router is outside the start and end times specified in the certificate.

Skipping the AAA Check of the Certificate

If the communication with an AAA server is protected with a certificate, and you want to skip the AAA check of the certificate, use the **match certificate** command with the **skip authorization-check** keyword. For example, if a virtual private network (VPN) tunnel is configured so that all AAA traffic goes over that tunnel, and the tunnel is protected with a certificate, you can use the **match certificate** command with the **skip authorization-check** keyword to skip the certificate check so that the tunnel can be established.

The **match certificate**command and the **skip authorization-check** keyword should be configured after PKI integration with an AAA server is configured.

Note If the AAA server is available only via an IPSec connection, the AAA server cannot be contacted until after the IPSec connection is established. The IPSec connection cannot be "brought up" because the certificate of the AAA server is not yet valid.

PKI Certificate Chain Validation

A certificate chain establishes a sequence of trusted certificates --from a peer certificate to the root CA certificate. Within a PKI hierarchy, all enrolled peers can validate the certificate of one another if the peers share a trusted root CA certificate or a common subordinate CA. Each CA corresponds to a trustpoint.

When a certificate chain is received from a peer, the default processing of a certificate chain path continues until the first trusted certificate, or trustpoint, is reached. In Cisco IOS Release 12.4(6)T and later releases, an administrator may configure the level to which a certificate chain is processed on all certificates including subordinate CA certificates.

Configuring the level to which a certificate chain is processed allows for the reauthentication of trusted certificates, the extension of a trusted certificate chain, and the completion of a certificate chain that contains a gap.

Reauthentication of Trusted Certificates

The default behavior is for the router to remove any trusted certificates from the certificate chain sent by the peer before the chain is validated. An administrator may configure certificate chain path processing so that the router does not remove CA certificates that are already trusted before chain validation, so that all certificates in the chain are re-authenticated for the current session.

Extending the Trusted Certificate Chain

The default behavior is for the router to use its trusted certificates to extend the certificate chain if there are any missing certificates in the certificate chain sent by the peer. The router will validate only certificates in the chain sent by the peer. An administrator may configure certificate chain path processing so that the certificates in the peer's certificate chain and the router's trusted certificates are validated to a specified point.

Completing Gaps in a Certificate Chain

An administrator may configure certificate chain processing so that if there is a gap in the configured Cisco IOS trustpoint hierarchy, certificates sent by the peer can be used to complete the set of certificates to be validated.

Note If the trustpoint is configured to require parent validation and the peer does not provide the full certificate chain, the gap cannot be completed and the certificate chain is rejected and invalid.

Note It is a configuration error if the trustpoint is configured to require parent validation and there is no parent trustpoint configured. The resulting certificate chain gap cannot be completed and the subordinate CA certificate cannot be validated. The certificate chain is invalid.

How to Configure Authorization and Revocation of Certificates for Your PKI

Configuring PKI Integration with a AAA Server

Perform this task to generate a AAA username from the certificate presented by the peer and specify which fields within a certificate should be used to build the AAA database username.

Note The following restrictions should be considered when using the **all** keyword as the subject name for the **authorization username** command:

- Some AAA servers limit the length of the username (for example, to 64 characters). As a result, the entire certificate subject name cannot be longer than the limitation of the server.
- Some AAA servers limit the available character set that may be used for the username (for example, a space [ ] and an equal sign [=] may not be acceptable). You cannot use the **all** keyword for a AAA server having such a character-set limitation.
- The **subject-name** command in the trustpoint configuration may not always be the final AAA subject name. If the fully qualified domain name (FQDN), serial number, or IP address of the router are included in a certificate request, the subject name field of the issued certificate will also have these components. To turn off the components, use the **fqdn**, **serial-number**, and **ip-address** commands with the **none** keyword.
- CA servers sometimes change the requested subject name field when they issue a certificate. For example, CA servers of some vendors switch the relative distinguished names (RDNs) in the requested subject names to the following order: CN, OU, O, L, ST, and C. However, another CA server might append the configured LDAP directory root (for example, O=cisco.com) to the end of the requested subject name.
- Depending on the tools you choose for displaying a certificate, the printed order of the RDNs in the subject name could be different. Cisco IOS software always displays the least significant RDN first, but other software, such as Open Source Secure Socket Layer (OpenSSL), does the opposite. Therefore, if you are configuring a AAA server with a full distinguished name (DN) (subject name) as the corresponding username, ensure that the Cisco IOS software style (that is, with the least significant RDN first) is used.

or

radius-server host *hostname* [**key** *string*]

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa new-model**
4. **aaa authorization network** *listname* [*method*]
5. **crypto pki trustpoint** *name*
6. **enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**]

7. revocation-check method
8. **exit**
9. **authorization username subjectname** *subjectname*
10. **authorization list** *listname*
11. **tacacs-server host** hostname [**key** string]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa new-model** **Example:** `Router(config)# aaa new-model`	Enables the AAA access control model.
Step 4	**aaa authorization network** *listname* [*method*] **Example:** `Router (config)# aaa authorization network maxaaa group tacacs+`	Sets the parameters that restrict user access to a network. • *method* --Can be **group radius**, **group tacacs+**, or **group group-name**.
Step 5	**crypto pki trustpoint** *name* **Example:** `Route (config)# crypto pki trustpoint msca`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

Step 6	**enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**]
	Example:
	`Router (ca-trustpoint)# enrollment url http://caserver.myexample.com`
	- or-
	`Router (ca-trustpoint)# enrollment url http://[2001:DB8:1:1::1]:80`
		Specifies the following enrollment parameters of the CA:
		• (Optional) The **mode** keyword specifies the registration authority (RA) mode, if your CA system provides an RA. By default, RA mode is disabled.
		• (Optional) The **retry period** keyword and *minutes* argument specifies the period, in minutes, in which the router waits before sending the CA another certificate request. Valid values are from 1 to 60. The default is 1.
		• (Optional) The **retry count** keyword and *number* argument specifies the number of times a router will resend a certificate request when it does not receive

	Command or Action	Purpose
		a response from the previous request. Valid values are from 1 to 100. The default is 10. • The *url* argument is the URL of the CA to which your router should send certificate requests. **Note** With the introduction of Cisco IOS Release 15.2(1)T, an IPv6 address can be added to the **http:** enrolment method. For example: http://[ipv6-address]:80. The IPv6 address must be enclosed in brackets in the URL. See the Command Reference document for more information on the other enrollment methods that can be used. • (Optional) The **pem** keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.
Step 7	revocation-check method **Example:** `Router (ca-trustpoint)# revocation-check crl`	(Optional) Checks the revocation status of a certificate.
Step 8	**exit** **Example:** `Router (ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 9	**authorization username subjectname** *subjectname* **Example:** `Router (config)# authorization username subjectname serialnumber`	Sets parameters for the different certificate fields that are used to build the AAA username. The *subjectname* argument can be any of the following: • **all** --Entire distinguished name (subject name) of the certificate. • **commonname** --Certification common name. • **country** --Certificate country. • **email** --Certificate e-mail. • **ipaddress** --Certificate IP address. • **locality** --Certificate locality. • **organization** --Certificate organization. • **organizationalunit** --Certificate organizational unit. • **postalcode** --Certificate postal code. • **serialnumber** --Certificate serial number.

	Command or Action	Purpose
		• **state** --Certificate state field. • **streetaddress** --Certificate street address. • **title** --Certificate title. • **unstructuredname** --Certificate unstructured name.
Step 10	**authorization list** *listname* **Example:** `Route (config)# authorization list maxaaa`	Specifies the AAA authorization list.

Step 11	**tacacs-server host** hostname [**key** string]
	Example:
	`Router(config)# tacacs-server host 192.0.2.2 key a_secret_key`
	Example:
	`radius-server host hostname [key string]`
	Example:
	`Router(config)# radius-server host 192.0.2.1 key another_secret_key`
		Specifies a TACACS+ host.
		or
		Specifies a RADIUS host.

Troubleshooting Tips

To display debug messages for the trace of interaction (message type) between the CA and the router, use the **debug crypto pki transactions**command. (See the sample output, which shows a successful PKI integration with AAA server exchange and a failed PKI integration with AAA server exchange.)

Successful Exchange

```
Router# debug crypto pki transactions
Apr 22 23:15:03.695: CRYPTO_PKI: Found a issuer match
Apr 22 23:15:03.955: CRYPTO_PKI: cert revocation status unknown.
Apr 22 23:15:03.955: CRYPTO_PKI: Certificate validated without revocation check
```

Each line that shows “CRYPTO_PKI_AAA” indicates the state of the AAA authorization checks. Each of the AAA AV pairs is indicated, and then the results of the authorization check are shown.

```
Apr 22 23:15:04.019: CRYPTO_PKI_AAA: checking AAA authorization (ipsecca_script_aaalist,
PKIAAA-L, <all>)
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint" = "CA1")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: reply attribute ("cert-serial" = "15DE")
Apr 22 23:15:04.503: CRYPTO_PKI_AAA: authorization passed
Apr 22 23:12:30.327: CRYPTO_PKI: Found a issuer match
```

Failed Exchange

```
Router# debug crypto pki transactions
Apr 22 23:11:13.703: CRYPTO_PKI_AAA: checking AAA authorization =
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-application" = “all”)
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint"= “CA1”)
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: reply attribute ("cert-serial" = “233D”)
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: parsed cert-lifetime-end as: 21:30:00
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: timezone specific extended
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: cert-lifetime-end is expired
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: cert-lifetime-end check failed.
Apr 22 23:11:14.203: CRYPTO_PKI_AAA: authorization failed
```

In the above failed exchange, the certificate has expired.

Configuring a Revocation Mechanism for PKI Certificate Status Checking

Perform this task to set up a CRL as the certificate revocation mechanism--CRLs or OCSP--that is used to check the status of certificates in a PKI.

The revocation-check Command

Use the **revocation-check** command to specify at least one method (OCSP, CRL, or skip the revocation check) that is to be used to ensure that the certificate of a peer has not been revoked. For multiple methods, the order in which the methods are applied is determined by the order specified via this command.

If your router does not have the applicable CRL and is unable to obtain one or if the OCSP server returns an error, your router will reject the peer’s certificate--unless you include the **none** keyword in your configuration. If the **none** keyword is configured, a revocation check will not be performed and the certificate will always be accepted.

Nonces and Peer Communications with OCSP Servers

When using OCSP, nonces, unique identifiers for OCSP requests, are sent by default during peer communications with your OCSP server. The use of nonces offers a more secure and reliable communication channel between the peer and OCSP server.

If your OCSP server does not support nonces, you may disable the sending of nonces. For more information, check your OCSP server documentation.

Before you begin

- Before issuing any client certificates, the appropriate settings on the server (such as setting the CDP) should be configured.
- When configuring an OCSP server to return the revocation status for a CA server, the OCSP server must be configured with an OCSP response signing certificate that is issued by that CA server. Ensure that the signing certificate is in the correct format, or the router will not accept the OCSP response. See your OCSP manual for additional information.

Note

- OCSP transports messages over HTTP, so there may be a time delay when you access the OCSP server.
- If the OCSP server depends on normal CRL processing to check revocation status, the same time delay that affects CRLs will also apply to OCSP.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. crypto pki trustpoint *name*
4. **ocsp url** *url*
5. **revocation-check** *method1* [*method2 method3*]]
6. **ocsp disable-nonce**
7. **exit**
8. **exit**
9. **show crypto pki certificates**
10. **show crypto pki trustpoints** [**status** | *label* [**status**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	crypto pki trustpoint *name* **Example:** `Router(config)# crypto pki trustpoint hazel`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

Step 4	**ocsp url** *url*
	Example:
	`Router(ca-trustpoint)# ocsp url http://ocsp-server`
	- or -
	`Router(ca-trustpoint)# ocsp url http://10.10.10.1:80`
		The *url* argument specifies the URL of an OCSP server so that the trustpoint can check the certificate status. This URL overrides the URL of the OCSP server (if one exists) in the Authority Info Access (AIA) extension of the certificate. All certificates associated with a configured trustpoint are checked by the OCSP server. The URL can be a hostname, IPv4 address, or an IPv6 address.

	Command or Action	Purpose
	- or - `Router(ca-trustpoint)# ocsp url http://[2001DB8:1:1::2]:80`	
Step 5	**revocation-check** *method1* [*method2 method3*]] **Example:** `Router(ca-trustpoint)# revocation-check ocsp none`	Checks the revocation status of a certificate. • **crl** --Certificate checking is performed by a CRL. This is the default option. • **none** --Certificate checking is ignored. • **ocsp** --Certificate checking is performed by an OCSP server. If a second and third method are specified, each method will be used only if the previous method returns an error, such as a server being down.
Step 6	**ocsp disable-nonce** **Example:** `Router(ca-trustpoint)# ocsp disable-nonce`	(Optional) Specifies that a nonce, or an OCSP request unique identifier, will not be sent during peer communications with the OCSP server.
Step 7	**exit** **Example:** `Router(ca-trustpoint)# exit`	Returns to global configuration mode.
Step 8	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.
Step 9	**show crypto pki certificates** **Example:** `Router# show crypto pki certificates`	(Optional) Displays information about your certificates.
Step 10	**show crypto pki trustpoints** [**status** | *label* [**status**]] **Example:** `Router# show crypto pki trustpoints`	Displays information about the trustpoint configured in router.

Configuring Certificate Authorization and Revocation Settings

Perform this task to specify a certificate-based ACL, to ignore revocation checks or expired certificates, to manually override the default CDP location, to manually override the OCSP server setting, to configure CRL caching, or to set session acceptance or rejection based on a certificate serial number, as appropriate.

Configuring Certificate-Based ACLs to Ignore Revocation Checks

To configure your router to use certificate-based ACLs to ignore revocation checks and expired certificates, perform the following steps:

- Identify an existing trustpoint or create a new trustpoint to be used when verifying the certificate of the peer. Authenticate the trustpoint if it has not already been authenticated. The router may enroll with this trustpoint if you want. Do not set optional CRLs for the trustpoint if you plan to use the **match certificate** command and **skip revocation-check** keyword.
- Determine the unique characteristics of the certificates that should not have their CRL checked and of the expired certificates that should be allowed.
- Define a certificate map to match the characteristics identified in the prior step.
- You can add the **match certificate** command and **skip revocation-check** keyword and the **match certificate command** and **allow expired-certificate** keyword to the trustpoint that was created or identified in the first step.

Note Certificate maps are checked even if the peer's public key is cached. For example, when the public key is cached by the peer, and a certificate map is added to the trustpoint to ban a certificate, the certificate map is effective. This prevents a client with the banned certificate, which was once connected in the past, from reconnecting.

Manually Overriding CDPs in a Certificate

Users can override the CDPs in a certificate with a manually configured CDP. Manually overriding the CDPs in a certificate can be advantageous when a particular server is unavailable for an extended period of time. The certificate's CDPs can be replaced with a URL or directory specification without reissuing all of the certificates that contain the original CDP.

Manually Overriding the OCSP Server Setting in a Certificate

Administrators can override the OCSP server setting specified in the Authority Information Access (AIA) field of the client certificate or set by the issuing the **ocsp url** command. One or more OCSP servers may be manually specified, either per client certificate or per group of client certificates by the **match certificate override ocsp** command. The **match certificate override ocsp**command overrides the client certificate AIA field or the **ocsp url**command setting if a client certificate is successfully matched to a certificate map during the revocation check.

Note Only one OCSP server can be specified per client certificate.

Configuring CRL Cache Control

By default, a new CRL will be downloaded after the currently cached CRL expires. Administrators can either configure the maximum amount of time in minutes a CRL remains in the cache by issuing the **crl cache delete-after** command or disable CRL caching by issuing the **crl cache none** command. Only the **crl-cache**

delete-aftercommand or the **crl-cache none** command may be specified. If both commands are entered for a trustpoint, the last command executed will take effect and a message will be displayed.

Neither the **crl-cache none** command nor the **crl-cache delete-after** command affects the currently cached CRL. If you configure the **crl-cache none** command, all CRLs downloaded after this command is issued will not be cached. If you configure the **crl-cache delete-after** command, the configured lifetime will only affect CRLs downloaded after this command is issued.

This functionality is useful is when a CA issues CRLs with no expiration date or with expiration dates days or weeks ahead.

Configuring Certificate Serial Number Session Control

A certificate serial number can be specified to allow a certificate validation request to be accepted or rejected by the trustpoint for a session. A session may be rejected, depending on certificate serial number session control, even if a certificate is still valid. Certificate serial number session control may be configured by using either a certificate map with the **serial-number** field or an AAA attribute, with the **cert-serial-not** command.

Using certificate maps for session control allows an administrator to specify a single certificate serial number. Using the AAA attribute allows an administrator to specify one or more certificate serial numbers for session control.

Before you begin

- The trustpoint should be defined and authenticated before attaching certificate maps to the trustpoint.
- The certificate map must be configured before the CDP override feature can be enabled or the **serial-number** command is issued.
- The PKI and AAA server integration must be successfully completed to use AAA attributes as described in “PKI and AAA Server Integration for Certificate Status.”

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. crypto pki certificate map label sequence-number
4. *field-name match-criteria match-value*
5. **exit**
6. **crypto pki trustpoint** *name*
7. Do one of the following:
   - **crl-cache none**
   - **crl-cache delete-after** *time*
8. **match certificate** *certificate-map-label* [**allow expired-certificate** | **skip revocation-check** | **skip authorization-check**
9. **match certificate** *certificate-map-label* **override cdp** {**url** | **directory**} *string*
10. **match certificate** *certificate-map-label* **override ocsp** [**trustpoint** *trustpoint-label*] *sequence-number* **url** *ocsp-url*
11. **exit**
12. **aaa new-model**

13. **aaa attribute list** *list-name*
14. **attribute type** {*name*}{*value*}
15. **exit**
16. **exit**
17. **show crypto pki certificates**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	crypto pki certificate map label sequence-number **Example:** `Router(config)# crypto pki certificate map Group 10`	Defines values in a certificate that should be matched or not matched and enters ca-certificate-map configuration mode.
Step 4	*field-name match-criteria match-value* **Example:** `Router(ca-certificate-map)# subject-name co MyExample`	Specifies one or more certificate fields together with their matching criteria and the value to match. The *field-name* is one of the following case-insensitive name strings or a date: • **alt-subject-name** • **expires-on** • **issuer-name** • **name** • **serial-number** • **subject-name** • **unstructured-subject-name** • **valid-start** **Note** Date field format is dd mm yyyy hh:mm:ss or mmm dd yyyy hh:mm:ss. The *match-criteria* is one of the following logical operators:

	Command or Action	Purpose
		• **co** --contains (valid only for name fields and serial number field) • **eq** --equal (valid for name, serial number, and date fields) • **ge** --greater than or equal (valid only for date fields) • **lt** --less than (valid only for date fields) • **nc** --does not contain (valid only for name fields and serial number field) • **ne** --not equal (valid for name, serial number, and date fields) The *match-value*is the name or date to test with the logical operator assigned by match-criteria. **Note** Use this command only when setting up a certificate-based ACL--not when setting up a certificate-based ACL to ignore revocation checks or expired certificates.
Step 5	**exit** **Example:** `Router(ca-certificate-map)# exit`	Returns to global configuration mode.
Step 6	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint Access2`	Declares the trustpoint, given name and enters ca-trustpoint configuration mode.

Step 7	Do one of the following:
	• **crl-cache none**
	• **crl-cache delete-after** *time*
	Example:
	`Router(ca-trustpoint)# crl-cache none`
	Example:
	`Router(ca-trustpoint)# crl-cache delete-after 20`
		(Optional) Disables CRL caching completely for all CRLs associated with the trustpoint.
		The **crl-cache none** command does not affect any currently cached CRLs. All CRLs downloaded after this command is configured will not be cached.
		(Optional) Specifies the maximum time CRLs will remain in the cache for all CRLs associated with the trustpoint.
		• *time* --The amount of time in minutes before the CRL is deleted.
		The **crl-cache delete-after** command does not affect any currently cached CRLs. The configured lifetime will only affect CRLs downloaded after this command is configured.

	Command or Action	Purpose
Step 8	**match certificate** *certificate-map-label* [**allow expired-certificate** | **skip revocation-check** | **skip authorization-check** **Example:** `Router(ca-trustpoint)# match certificate Group skip revocation-check`	(Optional) Associates the certificate-based ACL (that was defined via the **crypto pki certificate map** command) to a trustpoint. - *certificate-map-label* --Must match the *label* argument specified via the **crypto pki certificate map** command. - **allow expired-certificate** --Ignores expired certificates. - **skip revocation-check** --Allows a trustpoint to enforce CRLs except for specific certificates. - **skip authorization-check** --Skips the AAA check of a certificate when PKI integration with an AAA server is configured.
Step 9	**match certificate** *certificate-map-label* **override cdp** {**url** | **directory**} *string* **Example:** `Router(ca-trustpoint)# match certificate Group1 override cdp url http://server.cisco.com`	(Optional) Manually overrides the existing CDP entries for a certificate with a URL or directory specification. - *certificate-map-label* --A user-specified label that must match the *label* argument specified in a previously defined **crypto pki certificate map** command. - **url** --Specifies that the certificate's CDPs will be overridden with an HTTP or LDAP URL. - **directory** --Specifies that the certificate's CDPs will be overridden with an LDAP directory specification. - *string* --The URL or directory specification. **Note** Some applications may time out before all CDPs have been tried and will report an error message. The error message will not affect the router, and the Cisco IOS software will continue attempting to retrieve a CRL until all CDPs have been tried.
Step 10	**match certificate** *certificate-map-label* **override ocsp** [**trustpoint** *trustpoint-label*] *sequence-number* **url** *ocsp-url* **Example:** `Router(ca-trustpoint)# match certificate mycertmapname override ocsp trustpoint mytp 15 url http://192.0.2.2`	(Optional) Specifies an OCSP server, either per client certificate or per group of client certificates, and may be issued more than once to specify additional OCSP servers and client certificate settings including alternative PKI hierarchies. - *certificate-map-label* --The name of an existing certificate map. - **trustpoint** --The trustpoint to be used when validating the OCSP server certificate.

	Command or Action	Purpose
		• *sequence-number* --The order the **match certificate override ocsp** command statements apply to the certificate being verified. Matches are performed from the lowest sequence number to the highest sequence number. If more than one command is issued with the same sequence number, it overwrites the previous OCSP server override setting. • **url** --The URL of the OCSP server. When the certificate matches a configured certificate map, the AIA field of the client certificate and any previously issued **ocsp url** command settings are overwritten with the specified OCSP server. If no map-based match occurs, one of the following two cases will continue to apply to the client certificate. • If OCSP is specified as the revocation method, the AIA field value will continue to apply to the client certificate. • If the **ocsp url** configuration exists, the **ocsp url** configuration settings will continue to apply to the client certificates.
Step 11	**exit** **Example:** `Router(ca-trustpoint)# exit`	Returns to global configuration mode.
Step 12	**aaa new-model** **Example:** `Router(config)# aaa new-model`	(Optional) Enables the AAA access control model.
Step 13	**aaa attribute list** *list-name* **Example:** `Router(config)# aaa attribute list crl`	(Optional) Defines an AAA attribute list locally on a router and enters config-attr-list configuration mode.

Step 14	**attribute type** {*name*}{*value*}
	Example:
	`Router(config-attr-list)# attribute type cert-serial-not 6C4A`
		(Optional) Defines an AAA attribute type that is to be added to an AAA attribute list locally on a router.
		To configure certificate serial number session control, an administrator may specify a specific certificate in the *value* field to be accepted or rejected based on its serial number where *name* is set to **cert-serial-not**. If the serial number of the certificate matches the serial number specified by the attribute type setting, the certificate will be rejected.

	Command or Action	Purpose
		For a full list of available AAA attribute types, execute the **show aaa attributes** command.

Step 15	**exit**
	Example:
	`Router(ca-trustpoint)# exit`
	Example:
	`Router(config-attr-list)# exit`
		Returns to global configuration mode.

Step 16	**exit** **Example:** `Router(config)# exit`	Returns to privileged EXEC mode.
Step 17	**show crypto pki certificates** **Example:** `Router# show crypto pki certificates`	(Optional) Displays the components of the certificates installed on the router if the CA certificate has been authenticated.

Example

The following is a sample certificate. The OCSP-related extensions are shown using exclamation points.

```
Certificate:
        Data:
            Version: v3
            Serial Number:0x14
            Signature Algorithm:SHAwithRSA - 1.2.840.113549.1.1.4
            Issuer:CN=CA server,OU=PKI,O=Cisco Systems
            Validity:
                Not Before:Thursday, August 8, 2002 4:38:05 PM PST
                Not After:Tuesday, August 7, 2003 4:38:05 PM PST
            Subject:CN=OCSP server,OU=PKI,O=Cisco Systems
            Subject Public Key Info:
                Algorithm:RSA - 1.2.840.113549.1.1.1
                Public Key:
                    Exponent:65537
                    Public Key Modulus:(2048 bits) :
                        <snip>
            Extensions:
                Identifier:Subject Key Identifier - 2.5.29.14
                    Critical:no
                    Key Identifier:
                        <snip>
                Identifier:Authority Key Identifier - 2.5.29.35
                    Critical:no
                    Key Identifier:
                        <snip>
!                Identifier:OCSP NoCheck:- 1.3.6.1.5.5.7.48.1.5
                     Critical:no
                Identifier:Extended Key Usage:- 2.5.29.37
                     Critical:no
                     Extended Key Usage:
                     OCSPSigning
!
                Identifier:CRL Distribution Points - 2.5.29.31
                    Critical:no
                    Number of Points:1
                    Point 0
                        Distribution Point:
[URIName:ldap://CA-server/CN=CA server,OU=PKI,O=Cisco Systems]
        Signature:
            Algorithm:SHAwithRSA - 1.2.840.113549.1.1.4
            Signature:
            <snip>
```

The following example shows an excerpt of the running configuration output when adding a **match certificate override ocsp** command to the beginning of an existing sequence:

```
match certificate map3 override ocsp 5 url http://192.0.2.3/
show running-configuration
.
.
.
        match certificate map3 override ocsp 5 url http://192.0.2.3/
        match certificate map1 override ocsp 10 url http://192.0.2.1/
        match certificate map2 override ocsp 15 url http://192.0.2.2/
```

The following example shows an excerpt of the running configuration output when an existing **match certificate override ocsp** command is replaced and a trustpoint is specified to use an alternative PKI hierarchy:

```
match certificate map4 override ocsp trustpoint tp4 10 url http://192.0.2.4/newvalue
show running-configuration
.
.
.
        match certificate map3 override ocsp trustpoint tp3 5 url http://192.0.2.3/
        match certificate map1 override ocsp trustpoint tp1 10 url http://192.0.2.1/
        match certificate map4 override ocsp trustpoint tp4 10 url
http://192.0.2.4/newvalue
        match certificate map2 override ocsp trustpoint tp2 15 url http://192.0.2.2/
```

Troubleshooting Tips

If you ignored revocation check or expired certificates, you should carefully check your configuration. Verify that the certificate map properly matches either the certificate or certificates that should be allowed or the AAA checks that should be skipped. In a controlled environment, try modifying the certificate map and determine what is not working as expected.

Configuring Certificate Chain Validation

Perform this task to configure the processing level for the certificate chain path of your peer certificates.

Before you begin

- The device must be enrolled in your PKI hierarchy.

- The appropriate key pair must be associated with the certificate.

Note

- A trustpoint associated with the root CA cannot be configured to be validated to the next level.

The **chain-validation** command is configured with the **continue** keyword for the trustpoint associated with the root CA, an error message will be displayed and the chain validation will revert to the default **chain-validation**command setting.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. crypto pki trustpoint *name*
4. **chain-validation** [{**stop** | **continue**} [*parent-trustpoint*]]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	crypto pki trustpoint *name* **Example:** `Router(config)# crypto pki trustpoint ca-sub1`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**chain-validation** [{**stop** | **continue**} [*parent-trustpoint*]] **Example:** `Router(ca-trustpoint)# chain-validation continue ca-sub1`	Configures the level to which a certificate chain is processed on all certificates including subordinate CA certificates. • Use the **stop**keyword to specify that the certificate is already trusted. This is the default setting. • Use the **continue** keyword to specify that the that the subordinate CA certificate associated with the trustpoint must be validated. • The *parent-trustpoint* argument specifies the name of the parent trustpoint the certificate must be validated against.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Router(ca-trustpoint)# exit`	Returns to global configuration mode

Configuration Examples for Setting Up Authorization and Revocation of Certificates

Configuring and Verifying PKI AAA Authorization Examples

This section provides configuration examples of PKI AAA authorizations:

Router Configuration Example

The following **show running-config**command output shows the working configuration of a router that is set up to authorize VPN connections using the PKI Integration with AAA Server feature:

```
Router# show running-config
Building configuration...
!
version 12.3
!
hostname router7200router7200
!
aaa new-model
!
!
aaa authentication login default group tacacs+
aaa authentication login no_tacacs enable
aaa authentication ppp default group tacacs+
aaa authorization exec ACSLab group tacacs+
aaa authorization network ACSLab group tacacs+
aaa accounting exec ACSLab start-stop group tacacs+
aaa accounting network default start-stop group ACSLab
aaa session-id common
!
ip domain name example.com
!
crypto pki trustpoint EM-CERT-SERV
 enrollment url http://192.0.2.33:80
 serial-number
 crl optional
 rsakeypair STOREVPN 2048
 auto-enroll
 authorization list ACSLab
!
crypto pki certificate chain EM-CERT-SERV
 certificate 04
  30820214 3082017D A0030201 02020104 300D0609 2A864886 F70D0101 04050030
  17311530 13060355 0403130C 454D2D43 4552542D 53455256 301E170D 30343031
  31393232 30323535 5A170D30 35303131 38323230 3235355A 3030312E 300E0603
  55040513 07314437 45424434 301C0609 2A864886 F70D0109 02160F37 3230302D
  312E6772 696C2E63 6F6D3081 9F300D06 092A8648 86F70D01 01010500 03818D00
  30818902 818100BD F3B837AA D925F391 2B64DA14 9C2EA031 5A7203C4 92F8D6A8
  7D2357A6 BCC8596F A38A9B10 47435626 D59A8F2A 123195BB BE5A1E74 B1AA5AE0
  5CA162FF 8C3ACA4F B3EE9F27 8B031642 B618AE1B 40F2E3B4 F996BEFE 382C7283
  3792A369 236F8561 8748AA3F BC41F012 B859BD9C DB4F75EE 3CEE2829 704BD68F
  FD904043 0F555702 03010001 A3573055 30250603 551D1F04 1E301C30 1AA018A0
  16861468 7474703A 2F2F3633 2E323437 2E313037 2E393330 0B060355 1D0F0404
  030205A0 301F0603 551D2304 18301680 1420FC4B CF0B1C56 F5BD4C06 0AFD4E67
  341AE612 D1300D06 092A8648 86F70D01 01040500 03818100 79E97018 FB955108
  12F42A56 2A6384BC AC8E22FE F1D6187F DA5D6737 C0E241AC AAAEC75D 3C743F59
  08DEEFF2 0E813A73 D79E0FA9 D62DC20D 8E2798CD 2C1DC3EC 3B2505A1 3897330C
  15A60D5A 8A13F06D 51043D37 E56E45DF A65F43D7 4E836093 9689784D C45FD61D
  EC1F160C 1ABC8D03 49FB11B1 DA0BED6C 463E1090 F34C59E4
  quit
 certificate ca 01
  30820207 30820170 A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  17311530 13060355 0403130C 454D2D43 4552542D 53455256 301E170D 30333132
  31363231 34373432 5A170D30 36313231 35323134 3734325A 30173115 30130603
  55040313 0C454D2D 43455254 2D534552 5630819F 300D0609 2A864886 F70D0101
  01050003 818D0030 81890281 8100C14D 833641CF D784F516 DA6B50C0 7B3CB3C9
  589223AB 99A7DC14 04F74EF2 AAEEE8F5 E3BFAE97 F2F980F7 D889E6A1 2C726C69
  54A29870 7E7363FF 3CD1F991 F5A37CFF 3FFDD3D0 9E486C44 A2E34595 C2D078BB
  E9DE981E B733B868 AA8916C0 A8048607 D34B83C0 64BDC101 161FC103 13C06500
  22D6EE75 7D6CF133 7F1B515F 32830203 010001A3 63306130 0F060355 1D130101
  FF040530 030101FF 300E0603 551D0F01 01FF0404 03020186 301D0603 551D0E04
  16041420 FC4BCF0B 1C56F5BD 4C060AFD 4E67341A E612D130 1F060355 1D230418
  30168014 20FC4BCF 0B1C56F5 BD4C060A FD4E6734 1AE612D1 300D0609 2A864886
  F70D0101 04050003 81810085 D2E386F5 4107116B AD3AC990 CBE84063 5FB2A6B5
  BD572026 528E92ED 02F3A0AE 1803F2AE AA4C0ED2 0F59F18D 7B50264F 30442C41
  0AF19C4E 70BD3CB5 0ADD8DE8 8EF636BD 24410DF4 DB62DAFC 67DA6E58 3879AA3E
  12AFB1C3 2E27CB27 EC74E1FC AEE2F5CF AA80B439 615AA8D5 6D6DEDC3 7F9C2C79
  3963E363 F2989FB9 795BA8
  quit
!
!
crypto isakmp policy 10
 encr aes
 group 14
!
!
crypto ipsec transform-set ISC_TS_1 esp-aes esp-sha-hmac
!
crypto ipsec profile ISC_IPSEC_PROFILE_2
 set security-association lifetime kilobytes 530000000
 set security-association lifetime seconds 14400
 set transform-set ISC_TS_1
!
!
controller ISA 1/1
!
!
interface Tunnel0
 description MGRE Interface provisioned by ISC
 bandwidth 10000
 ip address 192.0.2.172 255.255.255.0
 no ip redirects
 ip mtu 1408
 ip nhrp map multicast dynamic
 ip nhrp network-id 101
 ip nhrp holdtime 500
 ip nhrp server-only
 no ip split-horizon eigrp 101
 tunnel source FastEthernet2/1
 tunnel mode gre multipoint
 tunnel key 101
 tunnel protection ipsec profile ISC_IPSEC_PROFILE_2
!
interface FastEthernet2/0
 ip address 192.0.2.1 255.255.255.0
 duplex auto
 speed auto
!
interface FastEthernet2/1
 ip address 192.0.2.2 255.255.255.0
 duplex auto
 speed auto
!
!
tacacs-server host 192.0.2.55 single-connection
tacacs-server directed-request
tacacs-server key company lab
!
ntp master 1
!
end
```

Debug of a Successful PKI AAA Authorization Example

The following **show debugging** command output shows a successful authorization using the PKI Integration with AAA Server feature:

```
Router# show debugging
General OS:
  TACACS access control debugging is on
  AAA Authentication debugging is on
  AAA Authorization debugging is on
Cryptographic Subsystem:
Crypto PKI Trans debugging is on
Router#
May 28 19:36:11.117: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:36:12.789: CRYPTO_PKI: Found a issuer match
May 28 19:36:12.805: CRYPTO_PKI: cert revocation status unknown.
May 28 19:36:12.805: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:36:12.813: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:36:12.813: AAA/BIND(00000042): Bind i/f
May 28 19:36:12.813: AAA/AUTHOR (0x42): Pick method list 'ACSLab'
May 28 19:36:12.813: TPLUS: Queuing AAA Authorization request 66 for processing
May 28 19:36:12.813: TPLUS: processing authorization request id 66
May 28 19:36:12.813: TPLUS: Protocol set to None .....Skipping
May 28 19:36:12.813: TPLUS: Sending AV service=pki
May 28 19:36:12.813: TPLUS: Authorization request created for 66(POD5.example.com)
May 28 19:36:12.813: TPLUS: Using server 192.0.2.55
May 28 19:36:12.813: TPLUS(00000042)/0/NB_WAIT/203A4628: Started 5 sec timeout
May 28 19:36:12.813: TPLUS(00000042)/0/NB_WAIT: wrote entire 46 bytes request
May 28 19:36:12.813: TPLUS: Would block while reading pak header
May 28 19:36:12.817: TPLUS(00000042)/0/READ: read entire 12 header bytes (expect 27 bytes)
May 28 19:36:12.817: TPLUS(00000042)/0/READ: read entire 39 bytes response
May 28 19:36:12.817: TPLUS(00000042)/0/203A4628: Processing the reply packet
May 28 19:36:12.817: TPLUS: Processed AV cert-application=all
May 28 19:36:12.817: TPLUS: received authorization response for 66: PASS
May 28 19:36:12.817: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
May 28 19:36:12.817: CRYPTO_PKI_AAA: authorization passed
Router#
Router#
May 28 19:36:18.681: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 101: Neighbor 192.0.2.171 (Tunnel0) is
 up: new adjacency
```

```
Router#
Router# show crypto isakmp sa
dst             src             state          conn-id slot
192.0.2.22     192.0.2.102   QM_IDLE             84    0
```

Debugs of a Failed PKI AAA Authorization Example

The following **show debugging** command output shows that the router is not authorized to connect using VPN. The messages are typical of those that you might see in such a situation.

In this example, the peer username was configured as not authorized, by moving the username to a Cisco Secure ACS group called VPN_Router_Disabled in Cisco Secure ACS. The router, router7200.example.com, has been configured to check with a Cisco Secure ACS AAA server prior to establishing a VPN connection to any peer.

```
Router# show debugging
General OS:
  TACACS access control debugging is on
  AAA Authentication debugging is on
  AAA Authorization debugging is on
Cryptographic Subsystem:
  Crypto PKI Trans debugging is on

Router#
May 28 19:48:29.837: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:48:31.509: CRYPTO_PKI: Found a issuer match
May 28 19:48:31.525: CRYPTO_PKI: cert revocation status unknown.
May 28 19:48:31.525: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:48:31.533: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:48:31.533: AAA/BIND(00000044): Bind i/f
May 28 19:48:31.533: AAA/AUTHOR (0x44): Pick method list 'ACSLab'
May 28 19:48:31.533: TPLUS: Queuing AAA Authorization request 68 for processing
May 28 19:48:31.533: TPLUS: processing authorization request id 68
May 28 19:48:31.533: TPLUS: Protocol set to None .....Skipping
May 28 19:48:31.533: TPLUS: Sending AV service=pki
May 28 19:48:31.533: TPLUS: Authorization request created for 68(POD5.example.com)
May 28 19:48:31.533: TPLUS: Using server 192.0.2.55
May 28 19:48:31.533: TPLUS(00000044)/0/NB_WAIT/203A4C50: Started 5 sec timeout
May 28 19:48:31.533: TPLUS(00000044)/0/NB_WAIT: wrote entire 46 bytes request
May 28 19:48:31.533: TPLUS: Would block while reading pak header
May 28 19:48:31.537: TPLUS(00000044)/0/READ: read entire 12 header bytes (expect 6 bytes)
May 28 19:48:31.537: TPLUS(00000044)/0/READ: read entire 18 bytes response
May 28 19:48:31.537: TPLUS(00000044)/0/203A4C50: Processing the reply packet
May 28 19:48:31.537: TPLUS: received authorization response for 68: FAIL
May 28 19:48:31.537: CRYPTO_PKI_AAA: authorization declined by AAA, or AAA server not found.
May 28 19:48:31.537: CRYPTO_PKI_AAA: No cert-application attribute found. Failing.
May 28 19:48:31.537: CRYPTO_PKI_AAA: authorization failed
May 28 19:48:31.537: CRYPTO_PKI: AAA authorization for list 'ACSLab', and user
'POD5.example.com' failed.
May 28 19:48:31.537: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.162 is
bad: certificate invalid
May 28 19:48:39.821: CRYPTO_PKI: Trust-Point EM-CERT-SERV picked up
May 28 19:48:41.481: CRYPTO_PKI: Found a issuer match
May 28 19:48:41.501: CRYPTO_PKI: cert revocation status unknown.
May 28 19:48:41.501: CRYPTO_PKI: Certificate validated without revocation check
May 28 19:48:41.505: CRYPTO_PKI_AAA: checking AAA authorization (ACSLab, POD5.example.com,
 <all>)
May 28 19:48:41.505: AAA/BIND(00000045): Bind i/f
May 28 19:48:41.505: AAA/AUTHOR (0x45): Pick method list 'ACSLab'
May 28 19:48:41.505: TPLUS: Queuing AAA Authorization request 69 for processing
May 28 19:48:41.505: TPLUS: processing authorization request id 69
May 28 19:48:41.505: TPLUS: Protocol set to None .....Skipping
May 28 19:48:41.505: TPLUS: Sending AV service=pki
May 28 19:48:41.505: TPLUS: Authorization request created for 69(POD5.example.com)
May 28 19:48:41.505: TPLUS: Using server 198.168.244.55
May 28 19:48:41.509: TPLUS(00000045)/0/IDLE/63B22834: got immediate connect on new 0
May 28 19:48:41.509: TPLUS(00000045)/0/WRITE/63B22834: Started 5 sec timeout
May 28 19:48:41.509: TPLUS(00000045)/0/WRITE: wrote entire 46 bytes request
May 28 19:48:41.509: TPLUS(00000045)/0/READ: read entire 12 header bytes (expect 6 bytes)
May 28 19:48:41.509: TPLUS(00000045)/0/READ: read entire 18 bytes response
May 28 19:48:41.509: TPLUS(00000045)/0/63B22834: Processing the reply packet
May 28 19:48:41.509: TPLUS: received authorization response for 69: FAIL
May 28 19:48:41.509: CRYPTO_PKI_AAA: authorization declined by AAA, or AAA server not found.
May 28 19:48:41.509: CRYPTO_PKI_AAA: No cert-application attribute found. Failing.
May 28 19:48:41.509: CRYPTO_PKI_AAA: authorization failed
May 28 19:48:41.509: CRYPTO_PKI: AAA authorization for list 'ACSLab', and user
'POD5.example.com' failed.
May 28 19:48:41.509: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.162 is
bad: certificate invalid
Router#
Router# show crypto iskmp sa
dst             src             state          conn-id slot
192.0.2.2        192.0.2.102   MM_KEY_EXCH         95    0
```

Configuring a Revocation Mechanism Examples

This section contains the following configuration examples that can be used when specifying a revocation mechanism for your PKI:

Configuring an OCSP Server Example

The following example shows how to configure the router to use the OCSP server that is specified in the AIA extension of the certificate:

```
Router(config)# crypto pki trustpoint mytp
Router(ca-trustpoint)# revocation-check ocsp
```

Specifying a CRL and Then an OCSP Server Example

The following example shows how to configure the router to download the CRL from the CDP. If the CRL is unavailable, the OCSP server that is specified in the AIA extension of the certificate will be used. If both options fail, certificate verification will also fail.

```
Router(config)# crypto pki trustpoint mytp
Router(ca-trustpoint)# revocation-check crl ocsp
```

Specifying an OCSP Server Example

The following example shows how to configure your router to use the OCSP server at the HTTP URL “http://myocspserver:81.” If the server is down, the revocation check will be ignored.

```
Router(config)# crypto pki trustpoint mytp
Router(ca-trustpoint)# ocsp url http://myocspserver:81
Router(ca-trustpoint)# revocation-check ocsp none
```

Disabling Nonces in Communications with the OCSP Server Example

The following example shows communications when a nonce, or a unique identifier for the OCSP request, is disabled for communications with the OCSP server:

```
Router(config)# crypto pki trustpoint mytp
Router(ca-trustpoint)# ocsp url http://myocspserver:81
Router(ca-trustpoint)# revocation-check ocsp none
Router(ca-trustpoint)# ocsp disable-nonce
```

Configuring a Hub Router at a Central Site for Certificate Revocation Checks Example

The following example shows a hub router at a central site that is providing connectivity for several branch offices to the central site.

The branch offices are also able to communicate directly with each other using additional IPSec tunnels between the branch offices.

The CA publishes CRLs on an HTTP server at the central site. The central site checks CRLs for each peer when setting up an IPSec tunnel with that peer.

The example does not show the IPSec configuration--only the PKI-related configuration is shown.

Home Office Hub Configuration

```
crypto pki trustpoint VPN-GW
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Central VPN Gateway
 revocation-check crl
```

Central Site Hub Router

```
Router# show crypto ca certificate
Certificate
  Status: Available
  Certificate Serial Number: 2F62BE14000000000CA0
  Certificate Usage: General Purpose
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    Name: Central VPN Gateway
    cn=Central VPN Gateway
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 00:43:26 GMT Sep 26 2003
    end   date: 00:53:26 GMT Sep 26 2004
    renew date: 00:00:00 GMT Jan 1 1970
  Associated Trustpoints: VPN-GW
CA Certificate
  Status: Available
  Certificate Serial Number: 1244325DE0369880465F977A18F61CA8
  Certificate Usage: Signature
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    cn=Central Certificate Authority
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 22:19:29 GMT Oct 31 2002
    end   date: 22:27:27 GMT Oct 31 2017
  Associated Trustpoints: VPN-GW
```

Trustpoint on the Branch Office Router

```
crypto pki trustpoint home-office
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none

ip-address none
 subject-name o=Home Office Inc,cn=Branch 1
 revocation-check crl
```

A certificate map is entered on the branch office router.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
branch1(config)# crypto pki certificate map central-site 10
branch1(ca-certificate-map)#
```

The output from the **show certificate** command on the central site hub router shows that the certificate was issued by the following:

```
cn=Central Certificate Authority
o=Home Office Inc
```

These two lines are combined into one line using a comma (,) to separate them, and the original lines are added as the first criteria for a match.

```
Router (ca-certificate-map)# issuer-name co cn=Central Certificate Authority, ou=Home Office
 Inc
!The above line wrapped but should be shown on one line with the line above it.
```

The same combination is done for the subject name from the certificate on the central site router (note that the line that begins with “Name:” is not part of the subject name and must be ignored when creating the certificate map criteria). This is the subject name to be used in the certificate map.

cn=Central VPN Gateway

o=Home Office Inc

```
Router (ca-certificate-map)# subject-name eq cn=central vpn gateway, o=home office inc
```

Now the certificate map is added to the trustpoint that was configured earlier.

```
Router (ca-certificate-map)# crypto pki trustpoint home-office
Router (ca-trustpoint)# match certificate central-site skip revocation-check
Router (ca-trustpoint)# exit
Router (config)# exit
```

The configuration is checked (most of configuration is not shown).

```
Router# write term
!Many lines left out
.
.
.
crypto pki trustpoint home-office
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Branch 1
 revocation-check crl
 match certificate central-site skip revocation-check
!
!
crypto pki certificate map central-site 10
 issuer-name co cn = Central Certificate Authority, ou = Home Office Inc
 subject-name eq cn = central vpn gateway, o = home office inc
!many lines left out
```

Note that the issuer-name and subject-name lines have been reformatted to make them consistent for later matching with the certificate of the peer.

If the branch office is checking the AAA, the trustpoint will have lines similar to the following:

```
crypto pki trustpoint home-office
 auth list allow_list
 auth user subj commonname
```

After the certificate map has been defined as was done above, the following command is added to the trustpoint to skip AAA checking for the central site hub.

```
match certificate central-site skip authorization-check
```

In both cases, the branch site router has to establish an IPSec tunnel to the central site to check CRLs or to contact the AAA server. However, without the **match certificate**command and **central-site skip authorization-check (argument and keyword)**, the branch office cannot establish the tunnel until it has checked the CRL or the AAA server. (The tunnel will not be established unless the **match certificate**command and **central-site skip authorization-check** argument and keyword are used.)

The **match certificate** command and **allow expired-certificate** keyword would be used at the central site if the router at a branch site had an expired certificate and it had to establish a tunnel to the central site to renew its certificate.

Trustpoint on the Central Site Router

```
crypto pki trustpoint VPN-GW
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Central VPN Gateway
 revocation-check crl
```

Trustpoint on the Branch 1 Site Router

```
Router# show crypto ca certificate
Certificate
  Status: Available
  Certificate Serial Number: 2F62BE14000000000CA0
  Certificate Usage: General Purpose
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    Name: Branch 1 Site
    cn=Branch 1 Site
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 00:43:26 GMT Sep 26 2003
    end   date: 00:53:26 GMT Oct 3 2003
    renew date: 00:00:00 GMT Jan 1 1970
  Associated Trustpoints: home-office
CA Certificate
  Status: Available
  Certificate Serial Number: 1244325DE0369880465F977A18F61CA8
  Certificate Usage: Signature
  Issuer:
    cn=Central Certificate Authority
    o=Home Office Inc
  Subject:
    cn=Central Certificate Authority
    o=Home Office Inc
  CRL Distribution Points:
    http://ca.home-office.com/CertEnroll/home-office.crl
  Validity Date:
    start date: 22:19:29 GMT Oct 31 2002
    end   date: 22:27:27 GMT Oct 31 2017
  Associated Trustpoints: home-office
```

A certificate map is entered on the central site router.

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router (config)# crypto pki certificate map branch1 10
Router (ca-certificate-map)# issuer-name co cn=Central Certificate Authority, ou=Home Office
 Inc
!The above line wrapped but should be part of the line above it.
Router (ca-certificate-map)# subject-name eq cn=Brahcn 1 Site,o=home office inc
```

The certificate map is added to the trustpoint.

```
Router (ca-certificate-map)# crypto pki trustpoint VPN-GW
Router (ca-trustpoint)# match certificate branch1 allow expired-certificate
Router (ca-trustpoint)# exit
Router (config) #exit
```

The configuration should be checked (most of the configuration is not shown).

```
Router# write term
!many lines left out
crypto pki trustpoint VPN-GW
 enrollment url http://ca.home-office.com:80/certsrv/mscep/mscep.dll
 serial-number none
 fqdn none
 ip-address none
 subject-name o=Home Office Inc,cn=Central VPN Gateway
 revocation-check crl
 match certificate branch1 allow expired-certificate
!
!
crypto pki certificate map central-site 10
 issuer-name co cn = Central Certificate Authority, ou = Home Office Inc
 subject-name eq cn = central vpn gateway, o = home office inc
! many lines left out
```

The **match certificate**command and **branch1 allow expired-certificate** (argument and keyword) and the certificate map should be removed as soon as the branch router has a new certificate.

Configuring Certificate Authorization and Revocation Settings Examples

This section contains the following configuration examples that can be used when specifying a CRL cache control setting or certificate serial number session control:

Configuring CRL Cache Control

The following example shows how to disable CRL caching for all CRLs associated with the CA1 trustpoint:

```
crypto pki trustpoint CA1
 enrollment url http://CA1:80
 ip-address FastEthernet0/0
 crl query ldap://ldap_CA1
 revocation-check crl
 crl-cache none
```

The current CRL is still cached immediately after executing the example configuration shown above:

Router# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 18:57:42 GMT Nov 26 2005
    NextUpdate: 22:57:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
      ldap://ldap.example.com/CN=name Cert Manager,O=example.com
```

When the current CRL expires, a new CRL is then downloaded to the router at the next update. The **crl-cache none**command takes effect and all CRLs for the trustpoint are no longer cached; caching is disabled. You can verify that no CRL is cached by executing the **show crypto pki crls** command. No output will be shown because there are no CRLs cached.

The following example shows how to configure the maximum lifetime of 2 minutes for all CRLs associated with the CA1 trustpoint:

```
crypto pki trustpoint CA1
 enrollment url http://CA1:80
 ip-address FastEthernet0/0
 crl query ldap://ldap_CA1
 revocation-check crl
 crl-cache delete-after 2
```

The current CRL is still cached immediately after executing the example configuration above for setting the maximum lifetime of a CRL:

Router# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 18:57:42 GMT Nov 26 2005
    NextUpdate: 22:57:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
      ldap://ldap.example.com/CN=name Cert Manager,O=example.com
When the current CRL expires, a new CRL is downloaded to the router at the next update and
 the crl-cache delete-after
command takes effect. This newly cached CRL and all subsequent CRLs will be deleted after
a maximum lifetime of 2 minutes.
You can verify that the CRL will be cached for 2 minutes by executing the show crypto pki
crls
 command. Note that the NextUpdate time is 2 minutes after the LastUpdate time.
```

Router# **show crypto pki crls**

```
CRL Issuer Name:
    cn=name Cert Manager,ou=pki,o=example.com,c=US
    LastUpdate: 22:57:42 GMT Nov 26 2005

    NextUpdate: 22:59:42 GMT Nov 26 2005
    Retrieved from CRL Distribution Point:
```

ldap://ldap.example.com/CN=name Cert Manager,O=example.com

Configuring Certificate Serial Number Session Control

The following example shows the configuration of certificate serial number session control using a certificate map for the CA1 trustpoint:

```
crypto pki trustpoint CA1
 enrollment url http://CA1
 chain-validation stop
 crl query ldap://ldap_server
 revocation-check crl
 match certificate crl
!
crypto pki certificate map crl 10
 serial-number co 279d
```

Note If the *match-criteria* value is set to **eq** (equal) instead of **co** (contains), the serial number must match the certificate map serial number exactly, including any spaces.

The following example shows the configuration of certificate serial number session control using AAA attributes. In this case, all valid certificates will be accepted if the certificate does not have the serial number “4ACA.”

```
crypto pki trustpoint CA1
 enrollment url http://CA1
 ip-address FastEthernet0/0
 crl query ldap://ldap_CA1
 revocation-check crl
 aaa new-model
!
```

```
aaa attribute list crl
attribute-type aaa-cert-serial-not 4ACA
```

The server log shows that the certificate with the serial number “4ACA” was rejected. The certificate rejection is shown using exclamation points.

```
.
.
.
Dec 3 04:24:39.051: CRYPTO_PKI: Trust-Point CA1 picked up
Dec 3 04:24:39.051: CRYPTO_PKI: locked trustpoint CA1, refcount is 1
Dec 3 04:24:39.051: CRYPTO_PKI: unlocked trustpoint CA1, refcount is 0
Dec 3 04:24:39.051: CRYPTO_PKI: locked trustpoint CA1, refcount is 1
Dec 3 04:24:39.135: CRYPTO_PKI: validation path has 1 certs
Dec 3 04:24:39.135: CRYPTO_PKI: Found a issuer match
Dec 3 04:24:39.135: CRYPTO_PKI: Using CA1 to validate certificate
Dec 3 04:24:39.135: CRYPTO_PKI: Certificate validated without revocation check
Dec 3 04:24:39.135: CRYPTO_PKI: Selected AAA username: 'PKIAAA'
Dec 3 04:24:39.135: CRYPTO_PKI: Anticipate checking AAA list:'CRL'
Dec 3 04:24:39.135: CRYPTO_PKI_AAA: checking AAA authorization (CRL, PKIAAA-L1, <all>)
Dec 3 04:24:39.135: CRYPTO_PKI_AAA: pre-authorization chain validation status (0x4)
Dec 3 04:24:39.135: AAA/BIND(00000021): Bind i/f
Dec 3 04:24:39.135: AAA/AUTHOR (0x21): Pick method list 'CRL'
.
.
.
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-application" = "all")
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-trustpoint" = "CA1")
!
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: reply attribute ("cert-serial-not" = "4ACA")
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: cert-serial doesn't match ("4ACA" != "4ACA")
!
Dec 3 04:24:39.175: CRYPTO_PKI_AAA: post-authorization chain validation status (0x7)
!
Dec 3 04:24:39.175: CRYPTO_PKI: AAA authorization for list 'CRL', and user 'PKIAAA' failed.
Dec 3 04:24:39.175: CRYPTO_PKI: chain cert was anchored to trustpoint CA1, and chain
validation result was: CRYPTO_PKI_CERT_NOT_AUTHORIZED
!
Dec 3 04:24:39.175: %CRYPTO-5-IKMP_INVAL_CERT: Certificate received from 192.0.2.43 is bad:
 certificate invalid
Dec 3 04:24:39.175: %CRYPTO-6-IKMP_MODE_FAILURE: Processing of Main mode failed with peer
at 192.0.2.43
.
.
.
```

Configuring Certificate Chain Validation Examples

This section contains the following configuration examples that can be used to specify the level of certificate chain processing for your device certificates:

Configuring Certificate Chain Validation from Peer to Root CA

In the following configuration example, all of the certificates will be validated--the peer, SubCA11, SubCA1, and RootCA certificates.

```
crypto pki trustpoint RootCA
 enrollment terminal
 chain-validation stop
 revocation-check none
 rsakeypair RootCA
crypto pki trustpoint SubCA1
 enrollment terminal
 chain-validation continue RootCA
 revocation-check none
 rsakeypair SubCA1
crypto pki trustpoint SubCA11
 enrollment terminal
 chain-validation continue SubCA1
 revocation-check none
 rsakeypair SubCA11
```

Configuring Certificate Chain Validation from Peer to Subordinate CA

In the following configuration example, the following certificates will be validated--the peer and SubCA1 certificates.

```
crypto pki trustpoint RootCA
 enrollment terminal
 chain-validation stop
 revocation-check none
 rsakeypair RootCA
crypto pki trustpoint SubCA1
 enrollment terminal
 chain-validation continue RootCA
 revocation-check none
 rsakeypair SubCA1
crypto pki trustpoint SubCA11
 enrollment terminal
 chain-validation continue SubCA1
 revocation-check none
 rsakeypair SubCA11
```

Configuring Certificate Chain Validation Through a Gap

In the following configuration example, SubCA1 is not in the configured Cisco IOS hierarchy but is expected to have been supplied in the certificate chain presented by the peer.

If the peer supplies the SubCA1 certificate in the presented certificate chain, the following certificates will be validated--the peer, SubCA11, and SubCA1 certificates.

If the peer does not supply the SubCA1 certificate in the presented certificate chain, the chain validation will fail.

```
crypto pki trustpoint RootCA
 enrollment terminal
 chain-validation stop
 revocation-check none
 rsakeypair RootCA
crypto pki trustpoint SubCA11
 enrollment terminal
 chain-validation continue RootCA
 revocation-check none
 rsakeypair SubCA11
```

Additional References

Related Documents

Related Topic	Document Title
PKI commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*
Overview of PKI, including RSA keys, certificate enrollment, and CAs	“Cisco IOS PKI Overview: Understanding and Planning a PKI” module
RSA key generation and deployment	“Deploying RSA Keys Within a PKI” module
Certificate enrollment: supported methods, enrollment profiles, configuration tasks	“Configuring Certificate Enrollment for a PKI” module
Cisco IOS certificate server overview information and configuration tasks	“Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment ” module
Recommended cryptographic algorithms	*Next Generation Encryption*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Certificate Authorization and Revocation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for PKI Certificate Authorization and Revocation

Feature Name	Releases	Feature Information
Cache Control Enhancements for Certification Revocation Lists		This feature provides users the ability to disable CRL caching or to specify the maximum lifetime for which a CRL will be cached in router memory. It also provides functionality to configure certificate serial number session control. The following commands were introduced or modified by this feature: **crl-cache delete-after, crl-cache none, crypto pki certificate map**
Certificate-Complete Chain Validation		This feature provides users the ability to configure the level to which a certificate chain is processed on all certificates including subordinate CA certificates. The following command was introduced by this feature: **chain-validation**
OCSP - Server Certification from Alternate Hierarchy		This feature provides users with the flexibility to specify multiple OCSP servers, either per client certificate or per group of client certificates, and provides the capability for OCSP server validation based on external CA certificates or self-signed certificates. The following command was introduced by this feature: **match certificate override ocsp**
Optional OCSP Nonce		This feature provides users with the ability to configure the sending of a nonce, or unique identifier for an OCSP request, during OCSP communications.
Certificate Security Attribute-Based Access Control		Under the IPsec protocol, CA interoperability permits Cisco IOS devices and a CA to communicate so that the Cisco IOS device can obtain and use digital certificates from the CA. Certificates contain several fields that are used to determine whether a device or user is authorized to perform a specified action. This feature adds fields to the certificate that allow specifying an ACL, creating a certificate-based ACL. The following commands were introduced or modified by this feature: **crypto pki certificate map**, **crypto pki trustpoint match certificate**
Online Certificate Status Protocol (OCSP)		This feature allows users to enable OCSP instead of CRLs to check certificate status. Unlike CRLs, which provide only periodic certificate status, OCSP can provide timely information regarding the status of a certificate. The following commands were introduced by this feature: **ocsp url**, **revocation-check**
PKI AAA Authorization Using the Entire Subject Name		This feature provides users with the ability to query the AAA server using the entire subject name from the certificate as a unique AAA username. The following command was modified by this feature: **authorization username**

Feature Name	Releases	Feature Information
PKI Integration with AAA Server		This feature provides additional scalability for authorization by generating a AAA username from the certificate presented by the peer. A AAA server is queried to determine whether the certificate is authorized for use by the internal component. The authorization is indicated by a component-specified label that must be present in the AV pair for the user. The following commands were introduced by this feature: **authorization list**, **authorization username**
PKI: Query Multiple Servers During Certificate Revocation Check		This feature introduces the ability for Cisco IOS software to make multiple attempts to retrieve the CRL, allowing operations to continue when a particular server is not available. In addition, the ability to override the CDPs in a certificate with a manually configured CDP has been introduced. Manually overriding the CDPs in a certificate can be advantageous when a particular server is unavailable for an extended period of time. The certificate's CDPs can be replaced with a URL or directory specification without reissuing all of the certificates that contain the original CDP. The following command was introduced by this feature: **match certificate override cdp**
Using Certificate ACLs to Ignore Revocation Check and Expired Certificates		This feature allows a certificate that meets specified criteria to be accepted regardless of the validity period of the certificate, or if the certificate meets the specified criteria, revocation checking does not have to be performed. Certificate ACLs are used to specify the criteria that the certificate must meet to be accepted or to avoid revocation checking. In addition, if AAA communication is protected by a certificate, this feature provides for the AAA checking of the certificate to be ignored. The following command was modified by this feature: **match certificate**
PKI High Availability		The following commands were introduced or modified: **crypto pki server**, **crypto pki server start**, **crypto pki server stop**, **crypto pki trustpoint**, **crypto key generate rsa**, **crypto key import pem**,**crypto key move rsa**, **show crypto key mypubkey rsa**.

CHAPTER 5

Configuring Certificate Enrollment for a PKI

This module describes the different methods available for certificate enrollment and how to set up each method for a participating PKI peer. Certificate enrollment, which is the process of obtaining a certificate from a certification authority (CA), occurs between the end host that requests the certificate and the CA. Each peer that participates in the public key infrastructure (PKI) must enroll with a CA.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for PKI Certificate Enrollment

Before configuring peers for certificate enrollment, you should have the following items:

- A generated Rivest, Shamir, and Adelman (RSA) key pair to enroll and a PKI in which to enroll.
- An authenticated CA.

- Familiarity with the module “Cisco IOS PKI Overview: Understanding and Planning a PKI.”
- Enable NTP on the device so that the PKI services such as auto enrollment and certificate rollover may function correctly.

Note As of Cisco IOS Release 12.3(7)T, all commands that begin with “**crypto ca**” have been changed to begin with “**crypto pki**.” Although the router will still accept **crypto ca** commands, all output will be be displayed **crypto pki**.

Information About Certificate Enrollment for a PKI

What Are CAs

A CA is an entity that issues digital certificates that other parties can use. It is an example of a trusted third party. CAs are characteristic of many PKI schemes.

A CA manages certificate requests and issues certificates to participating network devices. These services provide centralized key management for the participating devices to validate identities and to create digital certificates. Before any PKI operations can begin, the CA generates its own public key pair and creates a self-signed CA certificate; thereafter, the CA can sign certificate requests and begin peer enrollment for the PKI.

You can use the Cisco IOS certificate server or a CA provided by a third-party CA vendor.

Framework for Multiple CAs

A PKI can be set up in a hierarchical framework to support multiple CAs. At the top of the hierarchy is a root CA, which holds a self-signed certificate. The trust within the entire hierarchy is derived from the RSA key pair of the root CA. The subordinate CAs within the hierarchy can be enrolled with either the root CA or with another subordinate CA. Multiple tiers of CAs are configured by either the root CA or with another subordinate CA. Within a hierarchical PKI, all enrolled peers can validate the certificate of one another if the peers share a trusted root CA certificate or a common subordinate CA.

When to Use Multiple CAs

Multiple CAs provide users with added flexibility and reliability. For example, subordinate CAs can be placed in branch offices while the root CA is at the office headquarters. Also, different granting policies can be implemented per CA, so you can set up one CA to automatically grant certificate requests while another CA within the hierarchy requires each certificate request to be manually granted.

Scenarios in which at least a two-tier CA is recommended are as follows:

- Large and very active networks in which a large number of certificates are revoked and reissued. A multiple tier CA helps to control the size of the certificate revocation lists (CRLs).
- When online enrollment protocols are used, the root CA can be kept offline except to issue subordinate CA certificates. This scenario provides added security for the root CA.

Authentication of the CA

The certificate of the CA must be authenticated before the device will be issued its own certificate and before certificate enrollment can occur. Authentication of the CA typically occurs only when you initially configure PKI support at your router. To authenticate the CA, issue the **crypto pki authenticate** command, which authenticates the CA to your router by obtaining the self-signed certificate of the CA that contains the public key of the CA.

Note PKI does not support certificate with lifetime validity greater than the year 2099. So, It is recommended to choose a life time validity fewer than the value 2099.

Authentication via the fingerprint Command

Cisco IOS Release 12.3(12) and later releases allow you to issue the **fingerprint** command t o preenter a fingerprint that can be matched against the fingerprint of a CA certificate during authentication.

If a fingerprint is not preentered for a trustpoint, and if the authentication request is interactive, you must verify the fingerprint that is displayed during authentication of the CA certificate. If the authentication request is noninteractive, the certificate will be rejected without a preentered fingerprint.

Note If the authentication request is made using the command-line interface (CLI), the request is an interactive request. If the authentication request is made using HTTP or another management tool, the request is a noninteractive request.

Supported Certificate Enrollment Methods

Cisco IOS software supports the following methods to obtain a certificate from a CA:

- Simple Certificate Enrollment Protocol (SCEP)--A Cisco-developed enrollment protocol that uses HTTP to communicate with the CA or registration authority (RA). SCEP is the most commonly used method for sending and receiving requests and certificates.

Note To take advantage of automated certificate and key rollover functionality, you must be running a CA that supports rollover and SCEP must be used as your client enrollment method. If you are running a Cisco IOS CA, you must be running Cisco IOS Release 12.4(2)T or a later release for rollover support.

- PKCS12--The router imports certificates in PKCS12 format from an external server.
- IOS File System (IFS)--The router uses any file system that is supported by Cisco IOS software (such as TFTP, FTP, flash, and NVRAM) to send a certificate request and to receive the issued certificate. Users may enable IFS certificate enrollment when their CA does not support SCEP.

Note Prior to Cisco IOS Release 12.3(4)T, only the TFTP file system was supported within IFS.

- Manual cut-and-paste--The router displays the certificate request on the console terminal, allowing the user to enter the issued certificate on the console terminal. A user may manually cut-and-paste certificate requests and certificates when there is no network connection between the router and CA.
- Enrollment profiles-- Enrollment profiles are primarily used for EST or terminal based enrollment. In case that the CA server does not support SCEP, the recommended methods for enrollment are EST based enrollment or terminal based enrollment.
- Self-signed certificate enrollment for a trustpoint--The secure HTTP (HTTPS) server generates a self-signed certificate that is to be used during the secure socket layer (SSL) handshake, establishing a secure connection between the HTTPS server and the client. The self-signed certificate is then saved in the router's startup configuration (NVRAM). The saved, self-signed certificate can then be used for future SSL handshakes, eliminating the user intervention that was necessary to accept the certificate every time the router reloaded.

Note To take advantage of autoenrollment and autoreenrollment, do not use either TFTP or manual cut-and-paste enrollment as your enrollment method. Both TFTP and manual cut-and-paste enrollment methods are manual enrollment processes, requiring user input.

Cisco IOS Suite-B Support for Certificate Enrollment for a PKI

Suite-B requirements comprise of four user interface suites of cryptographic algorithms for use with IKE and IPSec that are described in RFC 4869. Each suite consists of an encryption algorithm, a digital signature algorithm, a key agreement algorithm, and a hash or message digest algorithm.

Suite-B adds the following support for the certificate enrollment for a PKI:

- Elliptic Curve Digital Signature Algorithm (ECDSA) (256-bit and 384-bit curves) is used for the signature operation within X.509 certificates.
- PKI support for validation of for X.509 certificates using ECDSA signatures.
- PKI support for generating certificate requests using ECDSA signatures and for importing the issued certificates into IOS.

See the Configuring Security for VPNs with IPsec feature module for more detailed information about Cisco IOS Suite-B support.

Registration Authorities

A Cisco IOS certificate server can be configured to run in RA mode. An RA offloads authentication and authorization responsibilities from a CA. When the RA receives a SCEP or manual enrollment request, the administrator can either reject or grant it on the basis of local policy. If the request is granted, it will be forwarded to the issuing CA, and the CA can be configured to automatically generate the certificate and return it to the RA. The client can later retrieve the granted certificate from the RA.

Automatic Certificate Enrollment

Automatic certificate enrollment allows the CA client to automatically request a certificate from its CA sever. This automatic router request eliminates the need for operator intervention when the enrollment request is sent to the CA server. Automatic enrollment is performed on startup for any trustpoint CA that is configured and that does not have a valid client certificate. When the certificate expires, a new certificate is automatically requested.

Note When automatic enrollment is configured, clients automatically request client certificates. The CA server performs its own authorization checks; if these checks include a policy to automatically issue certificates, all clients will automatically receive certificates, which is not very secure. Thus, automatic certificate enrollment should be combined with additional authentication and authorization mechanisms (such as Secure Device Provisioning (SDP), leveraging existing certificates, and one-time passwords).

Automated Client Certificate and Key Rollover

By default, the automatic certificate enrollment function requests a new client certificate and keys from the CS before the client’s current certificate expires. Certificate and key rollover allows the certificate renewal rollover request to be made before the certificate expires by retaining the current key and certificate until the new, or rollover, certificate is available. After a specified amount of time, the rollover certificate and keys will become the active certificate and keys. The expired certificate and keys are immediately deleted upon rollover and removed from the certificate chain and CRL.

The setup for automatic rollover is twofold: CA clients must be automatically enrolled and the client’s CAs must be automatically enrolled and have the **auto-rollover** command enabled. For more information on configuring your CA servers for automatic certificate rollover see the section “Automatic CA Certificate and Key Rollover” in the chapter “Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment ” of the *Public Key Infrastructure Configuration Guide*.

An optional renewal percentage parameter can be used with the **auto-enroll** command to allow a new certificate to be requested when a specified percentage of the lifetime of the certificate has passed. For example, if the renewal percentage is configured as 90 and the certificate has a lifetime of one year, a new certificate is requested 36.5 days before the old certificate expires. In order for automatic rollover to occur, the renewal percentage must be less than 100.The specified percent value must not be less than 10. If a client certificate is issued for less than the configured validity period due to the impending expiration of the CA certificate, the rollover certificate will be issued for the balance of that period. A minimum of 10 percent of the configured validity period, with an absolute minimum of 3 minutes, is required to allow rollover enough time to function.

Tip If CA autoenrollment is not enabled, you may manually initiate rollover on an existing client with the **crypto pki enroll** command if the expiration time of the current client certificate is equal to or greater than the expiration time of the corresponding CA certificate. The client will initiate the rollover process, which occurs only if the server is configured for automated rollover and has an available rollover server certificate.

Note A key pair is also sent if configured by the **auto-enroll re-generate** command and keyword. It is recommended that a new key pair be issued for security reasons.

Certificate Enrollment Profiles

Certificate enrollment profiles allow users to specify certificate authentication, enrollment, and reenrollment parameters when prompted. The values for these parameters are referenced by two templates that make up the profile. One template contains parameters for the HTTP request that is sent to the CA server to obtain the certificate of the CA (also known as certificate authentication); the other template contains parameters for the HTTP request that is sent to the CA for certificate enrollment.

Configuring two templates enables users to specify different URLs or methods for certificate authentication and enrollment; for example, authentication (getting the certificate of the CA) can be performed via TFTP (using the **authentication url** command) and enrollment can be performed manually (using the **enrollment terminal** command).

Prior to Cisco IOS Release 12.3(11)T, certificate requests could be sent only in a PKCS10 format; however, an additional parameter was added to the profile, allowing users to specify the PKCS7 format for certificate renewal requests.

Note A single enrollment profile can have up to three separate sections for each task--certificate authentication, enrollment, and reenrollment.

How to Configure Certificate Enrollment for a PKI

This section contains the following enrollment option procedures. If you configure enrollment or autoenrollment (the first task), you cannot configure manual certificate enrollment. Also, if you configure TFTP or manual cut-and-paste certificate enrollment, you cannot configure autoenrollment, autoreenrollment, an enrollment profile, nor can you utilize the automated CA certificate rollover capability.

Configuring Certificate Enrollment or Autoenrollment

Perform this task to configure certificate enrollment or autoenrollment for clients participating in your PKI.

Before you begin

Before configuring automatic certificate enrollment requests, you should ensure that all necessary enrollment information is configured.

Prerequisites for Enabling Automated Client Certificate and Key Rollover

CA client support for certificate rollover is automatically enabled when using autoenrollment. For automatic CA certificate rollover to run successfully, the following prerequisites are applicable:

- Your network devices must support shadow PKI.
- Your clients must be running Cisco IOS Release 12.4(2)T or a later release.
- The client’s CS must support automatic rollover. See the section “Automatic CA Certificate and Key Rollover” in the chapter “Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment ” of the *Public Key Infrastructure Configuration Guide* for more information on CA server automatic rollover configuration.

Prerequisites for Specifying Autoenrollment Initial Key Generation Location

To specify the location of the autoenrollment initial key generation, you must be running Cisco IOS Release 12.4(11)T or a later release.

RSA Key Pair Restriction for Autoenrollment

Trustpoints configured to generate a new key pair using the **regenerate** command or the **regenerate** keyword of the **auto-enroll** command must not share key pairs with other trustpoints. To give each trustpoint its own key pair, use the **rsakeypair** command in ca-trustpoint configuration mode. Sharing key pairs among regenerating trustpoints is not supported and will cause loss of service on some of the trustpoints because of key and certificate mismatches.

Certificate renewal with regenerate option does not work with key label starting from zero ('0'), for example, '0test'. CLI allows configuring such name under trustpoint, and allows hostname starting from zero, but certificate regenerate will fail.

Restrictions for Automated Client Certificate and Key Rollover

In order for clients to run automatic CA certificate rollover successfully, the following restrictions are applicable:

- SCEP must be used to support rollover. Any device that enrolls with the PKI using an alternative to SCEP as the certificate management protocol or mechanism (such as enrollment profiles, manual enrollment, or TFTP enrollment) will not be able to take advantage of the rollover functionality provided by SCEP.
- If the configuration cannot be saved to the startup configuration after a shadow certificate is generated, rollover will not occur.
- Rollover with key regenerate does not work when keypair name starts from zero ('0') (for example, '0test'). When configuring **rsakeypair** *name* under a trustpoint, do not configure name starting from zero. When keypair name is not configured and the default keypair is used, make sure the router hostname does not start from zero. If it does so, configure "**rsakeypair** *name* explicitly under the trustpoint with a different name.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment** [**mode** | **retry period** *minutes* | **retry count** *number*] **url** *url* [**pem**]
5. **eckeypair** *label*
6. **subject-name** [*x.500-name*]
7. **vrf** *vrf-name*
8. **ip-address** {*ip-address* | *interface* | **none**}
9. serial-number [none]
10. **auto-enroll** [*percent*] [**regenerate**]

11. **usage** *method1* [*method2* [*method3*]]
12. **password** *string*
13. **rsakeypair** *key-label* *key-size* *encryption-key-size*]]
14. **fingerprint** *ca-fingerprint*
15. **on** *devicename* **:**
16. **exit**
17. **crypto pki authenticate** *name*
18. **exit**
19. **copy system:running-config nvram:startup-config**
20. **show crypto pki certificates**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint mytp`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**enrollment** [**mode** | **retry period** *minutes* | **retry count** *number*] **url** *url* [**pem**] **Example:** `Router(ca-trustpoint)# enrollment url http://cat.example.com`	Specifies the URL of the CA on which your router should send certificate requests. • **mode** --Specifies RA mode if your CA system provides an RA. • **retry period** *minutes* --Specifies the wait period between certificate request retries. The default is 1 minute between retries. • **retry count** *number* -- Specifies the number of times a router will resend a certificate request when it does not receive a response from the previous request. (Specify from 1 to 100 retries.) • **url** *url* -- URL of the file system where your router should send certificate requests. An IPv6 address can be added in the URL enclosed in brackets. For example: http:// [2001:DB8:1:1::1]:80. • **pem** -- Adds privacy-enhanced mail (PEM) boundaries to the certificate request.

	Command or Action	Purpose
		Note An enrollment method other than TFTP or manual cut-and-paste must be configured to support autoenrollment.
Step 5	**eckeypair** *label* **Example:** `Router(ca-trustpoint)# eckeypair Router_1_Key`	(Optional) Configures the trustpoint to use an Elliptic Curve (EC) key on which certificate requests are generated using ECDSA signatures. The *label* argument specifies the EC key label that is configured using the **crypto key generate rsa** or **crypto key generate ec keysize** command in global configuration mode. See the Configuring Internet Key Exchange for IPsec VPNs feature module for more information. **Note** If an ECDSA signed certificate is imported without a trustpoint configuration, then the label defaults to the FQDN value.
Step 6	**subject-name** [*x.500-name*] **Example:** `Router(ca-trustpoint)# subject-name cat`	(Optional) Specifies the requested subject name that will be used in the certificate request. • *x.500-name* --If it is not specified, the fully qualified domain name (FQDN), which is the default subject name, will be used.
Step 7	**vrf** *vrf-name* **Example:** `Router(ca-trustpoint)# vrf myvrf`	(Optional) Specifies the the VRF instance in the public key infrastructure (PKI) trustpoint to be used for enrollment, certificate revocation list (CRL) retrieval, and online certificate status protocol (OCSP) status.

Step 8	**ip-address** {*ip-address* | *interface* | **none**}
	Example:
	`Router(ca-trustpoint)# ip address 192.168.1.66`
		(Optional) Includes the IP address of the specified interface in the certificate request.
		• Issue the *ip-address* argument to specify either an IPv4 or IPv6 address.
		• Issue the *interface* argument to specify an interface on the router.
		• Issue the **none** keyword if no IP address should be included.
		Note If this command is enabled, you will not be prompted for an IP address during enrollment for this trustpoint.

Step 9	serial-number [none] **Example:** `Router(ca-trustpoint)# serial-number`	(Optional) Specifies the router serial number in the certificate request, unless the **none** keyword is issued. • Issue the **none** keyword to specify that a serial number will not be included in the certificate request.

	Command or Action	Purpose
Step 10	**auto-enroll** [*percent*] [**regenerate**] **Example:** `Router(ca-trustpoint)# auto-enroll regenerate`	(Optional) Enables autoenrollment, allowing the client to automatically request a rollover certificate from the CA. • If autoenrollment is not enabled, the client must be manually re-enrolled in your PKI upon certificate expiration. • By default, only t he Domain Name System (DNS) name of the router is included in the certificate. • Use the *percent* argument to specify that a new certificate will be requested after the percentage of the lifetime of the current certificate is reached. • Use the **regenerate** keyword to generate a new key for the certificate even if a named key already exists. **Note** If the key pair being rolled over is exportable, the new key pair will also be exportable. The following comment will appear in the trustpoint configuration to indicate whether the key pair is exportable: “! RSA key pair associated with trustpoint is exportable.” **Note** It is recommended that a new key pair be generated for security reasons.
Step 11	**usage** *method1* [*method2* [*method3*]] **Example:** `Router(ca-trustpoint)# usage ssl-client`	(Optional) Specifies the intended use for the certificate. • Available options are **ike**, **ssl-client**, and **ssl-server**; the default is **ike**.
Step 12	**password** *string* **Example:** `Router(ca-trustpoint)# password string1`	(Optional) Specifies the revocation password for the certificate. • If this command is enabled, you will not be prompted for a password during enrollment for this trustpoint. **Note** When SCEP is used, this password can be used to authorize the certificate request--often via a one-time password or similar mechanism.
Step 13	**rsakeypair** *key-label* *key-size* *encryption-key-size*]] **Example:** `Router(ca-trustpoint)# rsakeypair key-label 2048 2048`	(Optional) Specifies which key pair to associate with the certificate. • A key pair with the *key-label* argument will be generated during enrollment if it does not already exist or if the **auto-enroll regenerate** command was issued. • Specify the *key-size* argument for generating the key, and specify the *encryption-key-size* argument to

	Command or Action	Purpose
		request separate encryption, signature keys, and certificates. The key-size and encryption-key-size must be the same size. Length of less than 2048 is not recommended. **Note** If this command is not enabled, the FQDN key pair is used.
Step 14	**fingerprint** *ca-fingerprint* **Example:** `Router(ca-trustpoint)# fingerprint 12EF53FA 355CD23E 12EF53FA 355CD23E`	(Optional) Specifies a fingerprint that can be matched against the fingerprint of a CA certificate during authentication. **Note** If the fingerprint is not provided and authentication of the CA certificate is interactive, the fingerprint will be displayed for verification.
Step 15	**on** *devicename* **:** **Example:** `Router(ca-trustpoint)# on usbtoken0:`	(Optional) Specifies that RSA keys will be created on the specified device upon autoenrollment initial key generation. • Devices that may be specified include NVRAM, local disks, and Universal Serial Bus (USB) tokens. USB tokens may be used as cryptographic devices in addition to a storage device. Using a USB token as a cryptographic device allows RSA operations such as key generation, signing, and authentication to be performed on the token.
Step 16	**exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 17	**crypto pki authenticate** *name* **Example:** `Router(config)# crypto pki authenticate mytp`	Retrieves the CA certificate and authenticates it. Check the certificate fingerprint if prompted. **Note** This command is optional if the CA certificate is already loaded into the configuration.
Step 18	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Step 19	**copy system:running-config nvram:startup-config**
	Example:
	`Router#`
	`copy system:running-config nvram:startup-config`
		(Optional) Copies the running configuration to the NVRAM startup configuration.
		Note Autoenrollment will not update NVRAM if the running configuration has been modified but not written to NVRAM.

	Command or Action	Purpose
Step 20	**show crypto pki certificates** **Example:** `Router# show crypto pki certificates`	(Optional) Displays information about your certificates, including any rollover certificates.

Configuring Manual Certificate Enrollment

Manual certificate enrollment can be set up via TFTP or the manual cut-and-paste method. Both options can be used if your CA does not support SCEP or if a network connection between the router and CA is not possible. Perform one of the following tasks to set up manual certificate enrollment:

PEM-Formatted Files for Certificate Enrollment Request

Using PEM-formatted files for certificate requests can be helpful for customers who are using terminal or profile-based enrollment to request certificates from their CA server. Customers using PEM-formatted files can directly use existing certificates on their routers.

Restrictions for Manual Certificate Enrollment

SCEP Restriction

We do not recommend switching URLs if SCEP is used; that is, if the enrollment URL is “http://myca,” do not change the enrollment URL after getting the CA certificate and before enrolling the certificate. A user can switch between TFTP and manual cut-and-paste.

Key Regeneration Restriction

Do not regenerate the keys manually using the **crypto key generate** command; key regeneration will occur when the **crypto pki enroll**command is issued if the **regenerate** keyword is specified.

Configuring Cut-and-Paste Certificate Enrollment

Perform this task to configure cut-and-paste certificate enrollment. This task helps you to configure manual certificate enrollment via the cut-and-paste method for peers participating in your PKI.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment terminal pem**
5. **fingerprint** *ca-fingerprint*
6. **exit**
7. **crypto pki authenticate** *name*
8. crypto pki enroll name
9. crypto pki import name certificate
10. **exit**

11. **show crypto pki certificates**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint mytp`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	**enrollment terminal pem** **Example:** `Router(ca-trustpoint)# enrollment terminal`	Specifies the manual cut-and-paste certificate enrollment method. • The certificate request will be displayed on the console terminal so that it may be manually copied (or cut). • **pem** --Configures the trustpoint to generate PEM-formatted certificate requests to the console terminal.
Step 5	**fingerprint** *ca-fingerprint* **Example:** `Router(ca-trustpoint)# fingerprint 12EF53FA 355CD23E 12EF53FA 355CD23E`	(Optional) Specifies a fingerprint that can be matched against the fingerprint of a CA certificate during authentication. **Note** If the fingerprint is not provided, it will be displayed for verification.
Step 6	**exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 7	**crypto pki authenticate** *name* **Example:** `Router(config)# crypto pki authenticate mytp`	Retrieves the CA certificate and authenticates it.
Step 8	crypto pki enroll name **Example:**	Generates certificate request and displays the request for copying and pasting into the certificate server.

	Command or Action	Purpose
	`Router(config)# crypto pki enroll mytp`	• You are prompted for enrollment information, such as whether to include the router FQDN and IP address in the certificate request. You are also given the choice about displaying the certificate request to the console terminal. • The base-64 encoded certificate with or without PEM headers as requested is displayed.
Step 9	crypto pki import name certificate **Example:** `Router(config)# crypto pki import mytp certificate`	Imports a certificate manually at the console terminal (pasting). • The base-64 encoded certificate is accepted from the console terminal and inserted into the internal certificate database. **Note** You must enter this command twice if usage keys, a signature key, and an encryption key are used. The first time the command is entered, one of the certificates is pasted into the router. The second time the command is entered, the other certificate is pasted into the router. It does not matter which certificate is pasted first. **Note** Some CAs ignore the usage key information in the certificate request and issue general purpose usage certificates. If this applies to the certificate authority you are using, import the general purpose certificate. The router will not use one of the two key pairs generated.
Step 10	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 11	**show crypto pki certificates** **Example:** `Router# show crypto pki certificates`	(Optional) Displays information about your certificates, the certificates of the CA, and RA certificates.

Configuring TFTP Certificate Enrollment

Perform this task to configure TFTP certificate enrollment. This task helps you to configure manual certificate enrollment using a TFTP server.

Before you begin

- You must know the correct URL to use if you are configuring certificate enrollment via TFTP.

- The router must be able to write a file to the TFTP server for the **crypto pki enroll** command.
- If you are using a file specification with the **enrollment** command, the file must contain the CA certificate either in binary format or be base-64 encoded.
- You must know if your CA ignores key usage information in a certificate request and issues only a general purpose usage certificate.

Caution Some TFTP servers require that the file must exist on the server before it can be written. Most TFTP servers require files that can be written over. This requirement may pose a risk because any router or other device may write or overwrite the certificate request; thus, the replacement certificate request will not be used by the CA administrator, who must first check the enrollment request fingerprint before granting the certificate request.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment** [**mode**] [**retry period** minutes] [**retry count** number] **url** url [**pem**]
5. **fingerprint** *ca-fingerprint*
6. **exit**
7. **crypto pki authenticate** *name*
8. crypto pki enroll name
9. crypto pki import name certificate
10. **exit**
11. **show crypto pki certificates**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint mytp`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

	Command or Action	Purpose
Step 4	**enrollment** [**mode**] [**retry period** minutes] [**retry count** number] **url** url [**pem**] **Example:** `Router(ca-trustpoint)# enrollment url tftp://certserver/file_specification`	Specifies TFTP as the enrollment method to send the enrollment request and to retrieve the CA certificate and router certificate and any optional parameters. **Note** For TFTP enrollment, the URL must be configured as a TFTP URL, tftp://example_tftp_url. • An optional file specification filename may be included in the TFTP URL. If the file specification is not included, the FQDN will be used. If the file specification is included, the router will append the extension ".ca" to the specified filename.
Step 5	**fingerprint** *ca-fingerprint* **Example:** `Router(ca-trustpoint)# fingerprint 12EF53FA 355CD23E 12EF53FA 355CD23E`	(Optional) Specifies the fingerprint of the CA certificate received via an out-of-band method from the CA administrator. **Note** If the fingerprint is not provided, it will be displayed for verification.
Step 6	**exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 7	**crypto pki authenticate** *name* **Example:** `Router(config)# crypto pki authenticate mytp`	Retrieves the CA certificate and authenticates it from the specified TFTP server.
Step 8	crypto pki enroll name **Example:** `Router(config)# crypto pki enroll mytp`	Generates certificate request and writes the request out to the TFTP server. • You are prompted for enrollment information, such as whether to include the router FQDN and IP address in the certificate request. You are queried about whether to display the certificate request to the console terminal. • The filename to be written is appended with the extension ".req". For usage keys, a signature key and an encryption key, two requests are generated and sent. The usage key request filenames are appended with the extensions "-sign.req" and "-encr.req", respectively.
Step 9	crypto pki import name certificate **Example:**	Imports a certificate via TFTP at the console terminal, which retrieves the granted certificate.

	Command or Action	Purpose
	`Router(config)# crypto pki import mytp certificate`	• The router will attempt to retrieve the granted certificate via TFTP using the same filename used to send the request, except the extension is changed from ".req" to ".crt". For usage key certificates, the extensions "-sign.crt" and "-encr.crt" are used. • The router will parse the received files, verify the certificates, and insert the certificates into the internal certificate database on the router. **Note** Some CAs ignore the usage key information in the certificate request and issue general purpose usage certificates. If your CA ignores the usage key information in the certificate request, only import the general purpose certificate. The router will not use one of the two key pairs generated.
Step 10	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 11	**show crypto pki certificates** **Example:** `Router# show crypto pki certificates`	(Optional) Displays information about your certificates, the certificates of the CA, and RA certificates.

Certifying a URL Link for Secure Communication with a Trend Micro Server

Perform this task to certify a link used in URL filtering that allows secure communication with a Trend Micro Server.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

SUMMARY STEPS

1. **enable**
2. **clock set** *hh* **:** *mm* **:** *ss* *date* *month* *year*
3. **configure terminal**
4. **clock timezone** *zone hours-offset* [*minutes-offset*]
5. **ip http server**
6. **hostname** *name*
7. **ip domain-name** *name*

8. **crypto key generate rsa general-keys modulus** *modulus-size*
9. **crypto pki trustpoint** *name*
10. **enrollment terminal**
11. **crypto ca authenticate** *name*
12. Copy the following block of text containing the base 64 encoded CA certificate and paste it at the prompt.
13. Enter **yes** to accept this certificate.
14. **serial-number**
15. **revocation-check none**
16. **end**
17. **trm register**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clock set** *hh* **:** *mm* **:** *ss date month year* **Example:** `Router# clock set 23:22:00 22 Dec 2009`	Sets the clock on the router.
Step 3	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 4	**clock timezone** *zone hours-offset* [*minutes-offset*] **Example:** `Router(config)# clock timezone PST -08`	Sets the time zone. • The *zone* argument is the name of the time zone (typically a standard acronym). The *hours-offset* argument is the number of hours the time zone is different from Universal Time Coordinated (UTC). The *minutes-offset* argument is the number of minutes the time zone is different from UTC. **Note** The *minutes-offset*argument of the **clock timezone** command is available for those cases where a local time zone is a percentage of an hour different from UTC or Greenwich Mean Time (GMT). For example, the time zone for some sections of Atlantic Canada (AST) is UTC-3.5. In this case, the necessary command would be **clock timezone AST -3 30**.

	Command or Action	Purpose
Step 5	**ip http server** **Example:** `Router(config)# ip http server`	Enables the HTTP server.
Step 6	**hostname** *name* **Example:** `Router(config)# hostname hostname1`	Configures the hostname of the router.
Step 7	**ip domain-name** *name* **Example:** `Router(config)# ip domain-name example.com`	Defines the domain name for the router.

Step 8	**crypto key generate rsa general-keys modulus** *modulus-size*
	Example:
	`Router(config)# crypto key generate rsa`
	`general-keys modulus general`
		Generates the crypto keys.
		• The **general-keys** keyword specifies that a general purpose key pair is generated, which is the default.
		• The **modulus** keyword and *modulus-size* argument specify the IP size of the key modulus. By default, the modulus of a CA key is 1024 bits. When generating RSA keys, you will be prompted to enter a modulus length. A longer modulus could offer stronger security but takes longer to generate and to use. A length of less than 2048 is not recommended.
		Note The name for the general keys that are generated are based on the domain name that is configured in Step 7. For example, the keys will be called "example.com."

Step 9	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint mytp`	Declares the CA that your router should use and enters ca-trustpoint configuration mode. **Note** Effective with Cisco IOS Release 12.3(8)T, the **crypto pki trustpoint** command replaced the **crypto ca trustpoint** command.
Step 10	**enrollment terminal** **Example:** `Router(ca-trustpoint)# enrollment terminal`	Specifies the manual cut-and-paste certificate enrollment method. • The certificate request will be displayed on the console terminal so that you may manually copy (or cut).
Step 11	**crypto ca authenticate** *name* **Example:**	Takes the name of the CA as the argument and authenticates it.

	Command or Action	Purpose

	`Router(ca-trustpoint)# crypto ca authenticate mytp`	• The following command output displays:
		`Enter the base 64 encoded CA certificate.`
		`End with a blank line or the word "quit" on a line`
		` by itself.`

Step 12	Copy the following block of text containing the base 64 encoded CA certificate and paste it at the prompt.	`MIIDIDCCAomgAwIBAgIENd70zzANBgkqhkiG9w0BAQUFADBOMQswCQYDVQQGEwJV`
		`UzEQMA4GA1UEChMHRXF1aWZheDEtMCsGA1UECxMkRXF1aWZheCBTZWN1cmUgQ2Vy`
		`dGlmaWNhdGUgQXV0aG9yaXR5MB4XDTk4MDgyMjE2NDE1MVoXDTE4MDgyMjE2NDE1`
		`MVowTjELMAkGA1UEBhMCVVMxEDAOBgNVBAoTB0VxdWlmYXgxLTArBgNVBAsTJEVx`
		`dWlmYXggU2VjdXJlIENlcnRpZmljYXRlIEF1dGhvcml0eTCBnzANBgkqhkiG9w0B`
		`AQEFAAOBjQAwgYkCgYEAwV2xWGcIYu6gmi0fCG2RFGiYCh7+2gRvE4RiIcPRfM6f`
		`BeC4AfBONOziipUEZKzxa1NfBbPLZ4C/QgKO/t0BCezhABRP/PvwDN1Dulsr4R+A`
		`cJkVV5MW8Q+XarfCaCMczE1ZMKxRHjuvK9buY0V7xdlfUNLjUA86iOe/FP3gx7kC`
		`AwEAAaOCAQkwggEFMHAGA1UdHwRpMGcwZaBjoGGkXzBdMQswCQYDVQQGEwJVUzEQ`
		`MA4GA1UEChMHRXF1aWZheDEtMCsGA1UECxMkRXF1aWZheCBTZWN1cmUgQ2VydGlm`
		`aWNhdGUgQXV0aG9yaXR5MQ0wCwYDVQQDEwRDUkwxMBoGA1UdEAQTMBGBDzIwMTgw`
		`ODIyMTY0MTUxWjALBgNVHQ8EBAMCAQYwHwYDVR0jBBgwFoAUSOZo+SvSspXXR9gj`
		`IBBPM5iQn9QwHQYDVR0OBBYEFEjmaPkr0rKV10fYIyAQTzOYkJ/UMAwGA1UdEwQF`
		`MAMBAf8wGgYJKoZIhvZ9B0EABA0wCxsFVjMuMGMDAgbAMA0GCSqGSIb3DQEBBQUA`
		`A4GBAFjOKer89961zgK5F7WF0bnj4JXMJTENAKaSbn+2kmOeUJXRmm/kEd5jhW6Y`
		`7qj/WsjTVbJmcVfewCHrPSqnI0kBBIZCe/zuf6IWUrVnZ9NA2zsmWLIodz2uFHdh`
		`1voqZiegDfqnc1zqcPGUIWVEX/r87yloqaKHee9570+sB3c4`
		The following command output displays:
		`Certificate has the following attributes:`
		`      Fingerprint MD5: 67CB9DC0 13248A82 9BB2171E`
		` D11BECD4`

	Command or Action	Purpose
		`    Fingerprint SHA1: D23209AD 23D31423 2174E40D 7F9D6213 9786633A`

Step 13	Enter **yes** to accept this certificate.	`% Do you accept this certificate? [yes/no]: yes`
		The following command output displays:
		`Trustpoint CA certificate accepted.`
		`% Certificate successfully imported`

Step 14	**serial-number** **Example:** `hostname1(ca-trustpoint)# serial-number`	Specifies the router serial number in the certificate request.
Step 15	**revocation-check none** **Example:** `hostname1(ca-trustpoint)# revocation-check none` **Example:**	Specifies that certificate checking is ignored.

Step 16	**end**
	Example:
	`hostname1(ca-trustpoint)# end`
		Exits ca-trustpoint configuration mode and returns to privileged EXEC mode.

Step 17	**trm register** **Example:** `hostname1# trm register`	Manually starts the Trend Micro Server registration process.

Configuring a Persistent Self-Signed Certificate for Enrollment via SSL

This section contains the following tasks:

Note These tasks are optional because if you enable the HTTPS server, it generates a self-signed certificate automatically using default values.

Persistent Self-Signed Certificates Overview

The SSL protocol can be used to establish a secure connection between an HTTPS server and a client (web browser). During the SSL handshake, the client expects the SSL server's certificate to be verifiable using a certificate the client already possesses.

If Cisco IOS software does not have a certificate that the HTTPS server can use, the server generates a self-signed certificate by calling a PKI application programming interface (API). When the client receives this self-signed certificate and is unable to verify it, intervention is needed. The client asks you if the certificate should be accepted and saved for future use. If you accept the certificate, the SSL handshake continues.

Future SSL handshakes between the same client and the server use the same certificate. However, if the router is reloaded, the self-signed certificate is lost. The HTTPS server must then create a new self-signed certificate. This new self-signed certificate does not match the previous certificate, so you are once again asked to accept it.

Requesting acceptance of the router's certificate each time that the router reloads may present an opportunity for an attacker to substitute an unauthorized certificate when you are being asked to accept the certificate. Persistent self-signed certificates overcome all these limitations by saving a certificate in the router's startup configuration.

Restrictions

- You can configure only one trustpoint for a persistent self-signed certificate.
- The maximum lifetime of a self-signed certificate is 00:00:00 GMT Jan 1, 2030.

Note Do not change the IP domain name or the hostname of the router after creating the self-signed certificate. Changing either name triggers the regeneration of the self-signed certificate and overrides the configured trustpoint. WebVPN ties the SSL trustpoint name to the WebVPN gateway configuration. If a new self-signed certificate is triggered, then the new trustpoint name does not match the WebVPN configuration, causing the WebVPN connections to fail.

Configuring a Trustpoint and Specifying Self-Signed Certificate Parameters

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

Perform the following task to configure a trustpoint and specify self-signed certificate parameters.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** name
4. **enrollment selfsigned**
5. **subject-name** [*x.500-name*]

6. **rsakeypair** *key-label* [key-size [encryption-key-size]]
7. **crypto pki enroll** name
8. **end**
9. **show crypto pki certificates** [*trustpoint-name*[**verbose**]]
10. **show crypto pki trustpoints** [**status** | *label* [**status**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** name **Example:** `Router(config)# crypto pki trustpoint local`	Declares the CA that your router should use and enters ca-trustpoint configuration mode. **Note** Effective with Cisco IOS Release 12.3(8)T, the **crypto pki trustpoint** command replaced the **crypto ca trustpoint** command.
Step 4	**enrollment selfsigned** **Example:** `Router(ca-trustpoint)# enrollment selfsigned`	Specifies self-signed enrollment.
Step 5	**subject-name** [*x.500-name*] **Example:** `Router(ca-trustpoint)# subject-name`	(Optional) Specifies the requested subject name to be used in the certificate request. • If no value for the *x-500-name* argument is specified, the FQDN, which is the default subject name, is used.
Step 6	**rsakeypair** *key-label* [key-size [encryption-key-size]] **Example:** `Router(ca-trustpoint)# rsakeypair examplekey 2048`	(Optional) Specifies which key pair to associate with the certificate. • The value for the *key-label* argument will be generated during enrollment if it does not already exist or if the **auto-enroll regenerate** command was issued. • Specify a value for the *key-size* argument for generating the key, and specify a value for the *encryption-key-size* argument to request separate encryption, signature keys, and certificates. The key-size and encryption-key-size must be the same size. Length of less than 2048 is no recommended.

	Command or Action	Purpose
		Note If this command is not enabled, the FQDN key pair is used.
Step 7	**crypto pki enroll** name **Example:** `Router(config)# crypto pki enroll local`	Tells the router to generate the persistent self-signed certificate.

Step 8	**end**
	Example:
	`Router(ca-trustpoint)# end`
		(Optional) Exits ca-trustpoint configuration mode.
		• Enter this command a second time to exit global configuration mode.

Step 9	**show crypto pki certificates** [*trustpoint-name*[**verbose**]] **Example:** `Router# show crypto pki certificates local verbose`	Displays information about your certificate, the certification authority certificate, and any registration authority certificates.
Step 10	**show crypto pki trustpoints** [**status** | *label* [**status**]] **Example:** `Router# show crypto pki trustpoints status`	Displays the trustpoints that are configured in the router.

Enabling the HTTPS Server

Perform the following task to enable the HTTPS server.

Before you begin

To specify parameters, you must create a trustpoint and configure it. To use default values, delete any existing self-signed trustpoints. Deleting all self-signed trustpoints causes the HTTPS server to generate a persistent self-signed certificate using default values as soon as the server is enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http secure-server**
4. **end**
5. **copy system:running-config nvram: startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip http secure-server** **Example:** `Router(config)# ip http secure-server`	Enables the HTTPS web server. **Note** A key pair (modulus 1024) and a self-signed certificate are automatically generated.

Step 4	**end**
	Example:
	`Router(config)# end`
		Exits global configuration mode.

Step 5	**copy system:running-config nvram: startup-config** **Example:** `Router# copy system:running-config nvram: startup-config`	Saves the self-signed certificate and the HTTPS server in enabled mode.

Configuring a Certificate Enrollment Profile for Enrollment or Reenrollment

Perform this task to configure a certificate enrollment profile for enrollment or reenrollment. This task helps you to configure an enrollment profile for certificate enrollment or reenrollment of a router with a Cisco IOS CA that is already enrolled with a third-party vendor CA.

Enable a router that is enrolled with a third-party vendor CA to use its existing certificate to enroll with the Cisco IOS certificate server so the enrollment request is automatically granted. To enable this functionality, you must issue the **enrollment credential** command. Also, you cannot configure manual certificate enrollment.

Before you begin

Perform the following tasks at the client router before configuring a certificate enrollment profile for the client router that is already enrolled with a third-party vendor CA so that the router can reenroll with a Cisco IOS certificate server:

- Defined a trustpoint that points to the third-party vendor CA.
- Authenticated and enrolled the client router with the third-party vendor CA.

Note

- To use certificate profiles, your network must have an HTTP interface to the CA.
- If an enrollment profile is specified, an enrollment URL may not be specified in the trustpoint configuration. Although both commands are supported, only one command can be used at a time in a trustpoint.
- Because there is no standard for the HTTP commands used by various CAs, the user is required to enter the command that is appropriate to the CA that is being used.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. enrollment profile label
5. **exit**
6. **crypto pki profile enrollment** *label*
7. Do one of the following:
   - **authentication url** *url*
   - **authentication terminal**
8. **authentication command**
9. Do one of the following:
   - **enrollment url** *url*
   -
   - **enrollment terminal**
10. **enrollment credential** *label*
11. **enrollment command**
12. **parameter** *number* {**value** *value* | **prompt** *string*}
13. **exit**
14. **show crypto pki certificates**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**crypto pki trustpoint** *name* **Example:** `Router(config)# crypto pki trustpoint Entrust`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.
Step 4	enrollment profile label **Example:** `Router(ca-trustpoint)# enrollment profile E`	Specifies that an enrollment profile is to be used for certificate authentication and enrollment.
Step 5	**exit** **Example:** `Router(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode.
Step 6	**crypto pki profile enrollment** *label* **Example:** `Router(config)# crypto pki profile enrollment E`	Defines an enrollment profile and enters ca-profile-enroll configuration mode. • *label* --Name for the enrollment profile; the enrollment profile name must match the name specified in the **enrollment profile** command.

Step 7	Do one of the following:
	• **authentication url** *url*
	• **authentication terminal**
	Example:
	`Router(ca-profile-enroll)# authentication url http://entrust:81`
	Example:
	`Router(ca-profile-enroll)# authentication terminal`
		Specifies the URL of the CA server to which to send certificate authentication requests.
		• *url* --URL of the CA server to which your router should send authentication requests. If you are using HTTP, the URL should read "http://CA_name," where CA_name is the host DNS name or IP address of the CA. If you are using TFTP, the URL should read "tftp://certserver/file_specification." (If the URL does not include a file specification, the FQDN of the router will be used.)
		Specifies manual cut-and-paste certificate authentication.

Step 8	**authentication command** **Example:** `Router(ca-profile-enroll)# authentication command`	(Optional) Specifies the HTTP command that is sent to the CA for authentication.
Step 9	Do one of the following: • **enrollment url** *url* • • **enrollment terminal** **Example:**	Specifies the URL of the CA server to which to send certificate enrollment requests via HTTP or TFTP. Specifies manual cut-and-paste certificate enrollment.

	Command or Action	Purpose

	`Router(ca-profile-enroll)# enrollment url http://entrust:81/cda-cgi/clientcgi.exe`
	Example:
	Example:
	`Router(ca-profile-enroll)# enrollment terminal`	

Step 10	**enrollment credential** *label* **Example:** `Router(ca-profile-enroll)# enrollment credential Entrust`	(Optional) Specifies the third-party vendor CA trustpoint that is to be enrolled with the Cisco IOS CA. **Note** This command cannot be issued if manual certificate enrollment is being used.
Step 11	**enrollment command** **Example:** `Router(ca-profile-enroll)# enrollment command`	(Optional) Specifies the HTTP command that is sent to the CA for enrollment.

Step 12	**parameter** *number* {**value** *value* | **prompt** *string*}
	Example:
	`Router(ca-profile-enroll)# parameter 1 value aaaa-bbbb-cccc`
		(Optional) Specifies parameters for an enrollment profile.
		• This command can be used multiple times to specify multiple values.

Step 13	**exit** **Example:** `Router(ca-profile-enroll)# exit`	(Optional) Exits ca-profile-enroll configuration mode. • Enter this command a second time to exit global configuration mode.
Step 14	**show crypto pki certificates** **Example:** `Router# show crypto pki certificates`	(Optional) Displays information about your certificates, the certificates of the CA, and RA certificates.

What to Do Next

If you configured the router to reenroll with a Cisco IOS CA, you should configure the Cisco IOS certificate server to accept enrollment requests only from clients already enrolled with the specified third-party vendor CA trustpoint to take advantage of this functionality. For more information, see the module “ Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment.”

Configuring Certificate Enrollment in a Two-Tier PKI Environment

The feature enables sub-CAs to issue certificates to their clients when a root CA is offline. The root certificate can be imported through the CLI first, and then it is used to validate the issuing sub CA certificate configured under the trustpoint.

Note Enable revocation checking as per your environment before performing the following tasks.

For importing the ROOT-CA through terminal, perform the following steps:

```
enable
!
configure terminal
!
crypto pki trustpoint ROOT-CA
revocation-check none
enrollment terminal
!
crypto pki authenticate ROOT-CA
!
exit
```

For authenticating SUB-CA without specifying or accepting the fingerprint.

```
enable
!
configure terminal
!
crypto pki trustpoint SUB-CA
revocation-check none
enrollment url url
chain-validation continue ROOT-CA
exit
!
crypto pki authenticate SUB-CA
exit
```

Configuration Examples for PKI Certificate Enrollment Requests

Configuring Certificate Enrollment or Autoenrollment Example

The following example shows the configuration for the “mytp-A” certificate server and its associated trustpoint, where RSA keys generated by the initial autoenrollment for the trustpoint will be stored on a USB token, “usbtoken0”:

```
crypto pki server mytp-A
   database level complete
   issuer-name CN=company, L=city, C=country
   grant auto
! Specifies that certificate requests will be granted automatically.
!
crypto pki trustpoint mytp-A
```

```
   revocation-check none
   rsakeypair myTP-A
   storage usbtoken0:
! Specifies that keys will be stored on usbtoken0:.
   on usbtoken0:
```

! Specifies that keys generated on initial auto enroll will be generated on and stored o**n !** usbtoken0:

Configuring Autoenrollment Example

The following example shows how to configure the router to automatically enroll with a CA on startup, enabling automatic rollover, and how to specify all necessary enrollment information in the configuration:

```
crypto pki trustpoint trustpt1
 enrollment url http://trustpt1.example.com//
 subject-name OU=Spiral Dept., O=example.com
 ip-address ethernet-0
 serial-number none
 usage ike
 auto-enroll regenerate
 password password1
 rsa-key trustpt1 2048
!
crypto pki certificate chain trustpt1
certificate pki 0B
30820293 3082023D A0030201 0202010B 300D0609 2A864886 F70D0101 04050030
79310B30 09060355 04061302 5553310B 30090603 55040813 02434131 15301306
0355040A 130C4369 73636F20 53797374 656D3120 301E0603 55040B13 17737562
6F726420 746F206B 6168756C 75692049 50495355 31243022 06035504 03131B79
6E692D75 31302043 65727469 66696361 7465204D 616E6167 6572301E 170D3030
30373134 32303536 32355A17 0D303130 37313430 31323834 335A3032 310E300C
06035504 0A130543 6973636F 3120301E 06092A86 4886F70D 01090216 11706B69
2D343562 2E636973 636F2E63 6F6D305C 300D0609 2A864886 F70D0101 01050003
4B003048 024100B3 0512A201 3B4243E1 378A9703 8AC5E3CE F77AF987 B5A422C4
15E947F6 70997393 70CF34D6 63A86B9C 4347A81A 0551FC02 ABA62360 01EF7DD2
6C136AEB 3C6C3902 03010001 A381F630 81F3300B 0603551D 0F040403 02052030
1C060355 1D110415 30138211 706B692D 3435622E 63697363 6F2E636F 6D301D06
03551D0E 04160414 247D9558 169B9A21 23D289CC 2DDA2A9A 4F77C616 301F0603
551D2304 18301680 14BD742C E892E819 1D551D91 683F6DB2 D8847A6C 73308185
0603551D 1F047E30 7C307AA0 3CA03AA4 38303631 0E300C06 0355040A 13054369
73636F31 24302206 03550403 131B796E 692D7531 30204365 72746966 69636174
65204D61 6E616765 72A23AA4 38303631 0E300C06 0355040A 13054369 73636F31
24302206 03550403 131B796E 692D7531 30204365 72746966 69636174 65204D61
6E616765 72300D06 092A8648 86F70D01 01040500 03410015 BC7CECF9 696697DF
E887007F 7A8DA24F 1ED5A785 C5C60452 47860061 0C18093D 08958A77 5737246B
0A25550A 25910E27 8B8B428E 32F8D948 3DD1784F 954C70
quit
```

Note In this example, keys are neither regenerated nor rolled over.

Configuring Certificate Autoenrollment with Key Regeneration Example

The following example shows how to configure the router to automatically enroll with the CA named “trustme1” on startup and enable automatic rollover. The **regenerate** keyword is issued, so a new key will be generated for the certificate and reissued when the automatic rollover process is initiated. The renewal percentage is configured as 90 so if the certificate has a lifetime of one year, a new certificate is requested 36.5 days before the old certificate expires. The changes made to the running configuration are saved to the NVRAM startup configuration because autoenrollment will not update NVRAM if the running configuration has been modified but not written to NVRAM.

```
crypto pki trustpoint trustme1
 enrollment url http://trustme1.example.com/
 subject-name OU=Spiral Dept., O=example.com
 ip-address ethernet0
 serial-number none
 auto-enroll 90 regenerate
 password password1
 rsakeypair trustme1 2048
 exit
crypto pki authenticate trustme1
copy system:running-config nvram:startup-config
```

Configuring Cut-and-Paste Certificate Enrollment Example

The following example shows how to configure certificate enrollment using the manual cut-and-paste enrollment method:

```
Router(config)#
crypto pki trustpoint TP
Router(ca-trustpoint)#
enrollment terminal
Router(ca-trustpoint)#
crypto pki authenticate TP
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
-----BEGIN CERTIFICATE-----
MIICNDCCAd6gAwIBAgIQOsCmXpVHwodKryRoqULV7jANBgkqhkiG9w0BAQUFADA5
MQswCQYDVQQGEwJVUzEWMBQGA1UEChMNQ2lzY28gU3lzdGVtczESMBAGA1UEAxMJ
bXNjYS1yb290MB4XDTAyMDIxNDAwNDYwMVoXDTA3MDIxNDAwNTQ0OFowOTELMAkG
A1UEBhMCVVMxFjAUBgNVBAoTDUNpc2NvIFN5c3RlbXMxEjAQBgNVBAMTCW1zY2Et
cm9vdDBcMA0GCSqGSIb3DQEBAQUAA0sAMEgCQQCix8nIGFg+wvy3BjFbVi25wYoG
K2N0HWWHpqxFuFhqyBnIC0OshIn9CtrdN3JvUNHr0NIKocEwNKUGYmPwWGTfAgMB
AAGjgcEwgb4wCwYDVR0PBAQDAgHGMA8GA1UdEwEB/wQFMAMBAf8wHQYDVR0OBBYE
FKIacsl6dKAfuNDVQymlSp7esf8jMG0GA1UdHwRmMGQwL6AtoCuGKWh0dHA6Ly9t
c2NhLXJvb3QvQ2VydEVucm9sbC9tc2NhLXJvb3QuY3JsMDGgL6AthitmaWxlOi8v
XFxtc2NhLXJvb3RcQ2VydEVucm9sbFxtc2NhLXJvb3QuY3JsMBAGCSsGAQQBgjcV
AQQDAgEAMA0GCSqGSIb3DQEBBQUAA0EAeuZkZMX9qkoLHfETYTpVWjZPQbBmwNRA
oJDSdYdtL3BcI/uLL5q7EmODyGfLyMGxuhQYx5r/40aSQgLCqBq+yg==
-----END CERTIFICATE-----
Certificate has the following attributes:
Fingerprint: D6C12961 CD78808A 4E02193C 0790082A
% Do you accept this certificate? [yes/no]:
y
Trustpoint CA certificate accepted.
% Certificate successfully imported
Router(config)#
crypto pki enroll TP
% Start certificate enrollment..
% The subject name in the certificate will be:
Router.example.com
% Include the router serial number in the subject name? [yes/no]:
n
% Include an IP address in the subject name? [no]:
n
Display Certificate Request to terminal? [yes/no]:
y
```

```
Signature key certificate request -
Certificate Request follows:
MIIBhTCB7wIBADAlMSMwIQYJKoZIhvcNAQkCFhRTYW5kQmFnZ2VyLmNpc2NvLmNv
bTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAxdhXFDiWAn/hIZs9zfOtssKA
daoWYu0ms9Fe/Pew01dh14vXdxgacstOs2Pr5wk6jLOPxpvxOJPWyQM6ipLmyVxv
ojhyLTrVohrh6Dnqcvk+G/5ohss9o9RxvONwx042pQchFnx9EkMuZC7evwRxJEqR
mBHXBZ8GmP3jYQsjS8MCAwEAAaAhMB8GCSqGSIb3DQEJDjESMBAwDgYDVR0PAQH/
BAQDAgeAMA0GCSqGSIb3DQEBBAUAA4GBAMT6WtyFw95POY7UtF+YIYHiVRUf4SCq
hRIAGrljUePLo9iTqyPU1Pnt8JnIZ5P5BHU3MfgP8sqodaWub6mubkzaohJ1qD06
O87fnLCNid5Tov5jKogFHIki2EGGZxBosUw9lJlenQdNdDPbJc5LIWdfDvciA6jO
Nl8rOtKnt8Q+
!
!
!
Redisplay enrollment request? [yes/no]:
Encryption key certificate request -
Certificate Request follows:
MIIBhTCB7wIBADAlMSMwIQYJKoZIhvcNAQkCFhRTYW5kQmFnZ2VyLmNpc2NvLmNv
bTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAwG60QojpDbzbKnyj8FyTiOcv
THkDP7XD4vLT1XaJ409z0gSIoGnIcdFtXhVlBWtpq3/O9zYFXr1tH+BMCRQi3Lts
0IpxYa3D9iFPqev7SPXpsAIsY8a6FMq7TiwLObqiQjLKL4cbuV0Frjl0Yuv5A/Z+
kqMOm7c+pWNWFdLe9lsCAwEAAaAhMB8GCSqGSIb3DQEJDjESMBAwDgYDVR0PAQH/
BAQDAgUgMA0GCSqGSIb3DQEBBAUAA4GBACF7feURj/fJMojPBlR6fa9BrlMJx+2F
H91YM/CIiz2n4mHTeWTWKhLoT8wUfa9NGOk7yi+nF/F7035twLfq6n2bSCTW4aem
8jLMMaeFxwkrV/ceQKrucmNC1uVx+fBy9rhnKx8j60XE25tnp1U08r6om/pBQABU
eNPFhozcaQ/2
!
!
!
Redisplay enrollment request? [yes/no]:
n
Router(config)#
crypto pki import TP certificate
Enter the base 64 encoded certificate.
End with a blank line or the word "quit" on a line by itself
MIIDajCCAxSgAwIBAgIKFN7C6QAAAAAMRzANBgkqhkiG9w0BAQUFADA5MQswCQYD
VQQGEwJVUzEWMBQGA1UEChMNQ2lzY28gU3lzdGVtczESMBAGA1UEAxMJbXNjYS1y
b290MB4XDTAyMDYwODAxMTY0MloXDTAzMDYwODAxMjY0MlowJTEjMCEGCSqGSIb3
DQEJAhMUU2FuZEJhZ2dlci5jaXNjby5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0A
MIGJAoGBAMXYVxQ4lgJ/4SGbPc3zrbLCgHWqFmLtJrPRXvz3sNNXYdeL13cYGnLL
TrNj6+cJOoyzj8ab8TiT1skDOoqS5slcb6I4ci061aIa4eg56nL5Phv+aIbLPaPU
cbzjcMdONqUHIRZ8fRJDLmQu3r8EcSRKkZgR1wWfBpj942ELI0vDAgMBAAGjggHM
MIIByDALBgNVHQ8EBAMCB4AwHQYDVR0OBBYEFL8Quz8dyz4EGIeKx9A8UMNHLE4s
MHAGA1UdIwRpMGeAFKIacsl6dKAfuNDVQymlSp7esf8joT2kOzA5MQswCQYDVQQG
EwJVUzEWMBQGA1UEChMNQ2lzY28gU3lzdGVtczESMBAGA1UEAxMJbXNjYS1yb290
ghA6wKZelUfCh0qvJGipQtXuMCIGA1UdEQEB/wQYMBaCFFNhbmRCYWdnZXIuY2lz
Y28uY29tMG0GA1UdHwRmMGQwL6AtoCuGKWh0dHA6Ly9tc2NhLXJvb3QvQ2VydEVu
cm9sbC9tc2NhLXJvb3QuY3JsMDGgL6AthitmaWxlOi8vXFxtc2NhLXJvb3RcQ2Vy
dEVucm9sbFxtc2NhLXJvb3QuY3JsMIGUBggrBgEFBQcBAQSBhzCBhDA/BggrBgEF
BQcwAoYzaHR0cDovL21zY2Etcm9vdC9DZXJ0RW5yb2xsL21zY2Etcm9vdF9tc2Nh
LXJvb3QuY3J0MEEGCCsGAQUFBzAChjVmaWxlOi8vXFxtc2NhLXJvb3RcQ2VydEVu
cm9sbFxtc2NhLXJvb3RfbXNjYS1yb290LmNydDANBgkqhkiG9w0BAQUFAANBAJo2
r6sHPGBdTQX2EDoJpR/A2UHXxRYqVSHkFKZw0z31r5JzUM0oPNUETV7mnZlYNVRZ
CSEX/G8boi3WOjz9wZo=
% Router Certificate successfully imported
Router(config)#
crypto pki import TP cert
Enter the base 64 encoded certificate.
End with a blank line or the word "quit" on a line by itself
MIIDajCCAxSgAwIBAgIKFN7OBQAAAAAMSDANBgkqhkiG9w0BAQUFADA5MQswCQYD
VQQGEwJVUzEWMBQGA1UEChMNQ2lzY28gU3lzdGVtczESMBAGA1UEAxMJbXNjYS1y
b290MB4XDTAyMDYwODAxMTY0NVoXDTAzMDYwODAxMjY0NVowJTEjMCEGCSqGSIb3
DQEJAhMUU2FuZEJhZ2dlci5jaXNjby5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0A
MIGJAoGBAMButEKI6Q282yp8o/Bck4jnL0x5Az+1w+Ly09V2ieNPc9IEiKBpyHHR
```

```
bV4VZQVraat/zvc2BV69bR/gTAkUIty7bNCKcWGtw/YhT6nr+0j16bACLGPGuhTK
u04sCzm6okIyyi+HG7ldBa45dGLr+QP2fpKjDpu3PqVjVhXS3vZbAgMBAAGjggHM
MIIByDALBgNVHQ8EBAMCBSAwHQYDVR0OBBYEFPDO29oRdlEUSgBMg6jZR+YFRWlj
MHAGA1UdIwRpMGeAFKIacsl6dKAfuNDVQymlSp7esf8joT2kOzA5MQswCQYDVQQG
EwJVUzEWMBQGA1UEChMNQ2lzY28gU3lzdGVtczESMBAGA1UEAxMJbXNjYS1yb290
ghA6wKZelUfCh0qvJGipQtXuMCIGA1UdEQEB/wQYMBaCFFNhbmRCYWdnZXIuY2lz
Y28uY29tMG0GA1UdHwRmMGQwL6AtoCuGKWh0dHA6Ly9tc2NhLXJvb3QvQ2VydEVu
cm9sbC9tc2NhLXJvb3QuY3JsMDGgL6AthitmaWxlOi8vXFxtc2NhLXJvb3RcQ2Vy
dEVucm9sbFxtc2NhLXJvb3QuY3JsMIGUBggrBgEFBQcBAQSBhzCBhDA/BggrBgEF
BQcwAoYzaHR0cDovL21zY2Etcm9vdC9DZXJ0RW5yb2xsL21zY2Etcm9vdF9tc2Nh
LXJvb3QuY3J0MEEGCCsGAQUFBzAChjVmaWxlOi8vXFxtc2NhLXJvb3RcQ2VydEVu
cm9sbFxtc2NhLXJvb3RfbXNjYS1yb290LmNydDANBgkqhkiG9w0BAQUFAANBAHaU
hyCwLirUghNxCmLzXRG7C3W1j0kSX7a4fX9OxKR/Z2SoMjdMNPPyApuh8SoT2zBP
ZKjZU2WjcZG/nZF4W5k=
% Router Certificate successfully imported
```

You can verify that the certificate was successfully imported by issuing the **show crypto pki certificates** command:

```
Router# show crypto pki certificates
Certificate
  Status: Available
  Certificate Serial Number: 14DECE05000000000C48
  Certificate Usage: Encryption
  Issuer:
    CN = TPCA-root
     O = Company
     C = US
  Subject:
    Name: Router.example.com
    OID.1.2.840.113549.1.9.2 = Router.example.com
  CRL Distribution Point:
    http://tpca-root/CertEnroll/tpca-root.crl
  Validity Date:
    start date: 18:16:45 PDT Jun 7 2002
    end   date: 18:26:45 PDT Jun 7 2003
    renew date: 16:00:00 PST Dec 31 1969
  Associated Trustpoints: TP
Certificate
  Status: Available
  Certificate Serial Number: 14DEC2E9000000000C47
  Certificate Usage: Signature
  Issuer:
    CN = tpca-root
     O = company
     C = US
  Subject:
    Name: Router.example.com
    OID.1.2.840.113549.1.9.2 = Router.example.com
  CRL Distribution Point:
    http://tpca-root/CertEnroll/tpca-root.crl
  Validity Date:
    start date: 18:16:42 PDT Jun 7 2002
    end   date: 18:26:42 PDT Jun 7 2003
    renew date: 16:00:00 PST Dec 31 1969
  Associated Trustpoints: TP
CA Certificate
  Status: Available
  Certificate Serial Number: 3AC0A65E9547C2874AAF2468A942D5EE
  Certificate Usage: Signature
  Issuer:
    CN = tpca-root
     O = Company
     C = US
  Subject:
    CN = tpca-root
     O = company
     C = US
  CRL Distribution Point:
    http://tpca-root/CertEnroll/tpca-root.crl
  Validity Date:
    start date: 16:46:01 PST Feb 13 2002
    end   date: 16:54:48 PST Feb 13 2007
  Associated Trustpoints: TP
```

Configuring Manual Certificate Enrollment with Key Regeneration Example

The following example shows how to regenerate new keys with a manual certificate enrollment from the CA named “trustme2”:

```
crypto pki trustpoint trustme2
 enrollment url http://trustme2.example.com/
 subject-name OU=Spiral Dept., O=example.com
 ip-address ethernet0
 serial-number none
 regenerate
 password password1
 rsakeypair trustme2 2048
 exit
crypto pki authenticate trustme2
crypto pki enroll trustme2
```

Creating and Verifying a Persistent Self-Signed Certificate Example

The following example shows how to declare and enroll a trustpoint named “local” and generate a self-signed certificate with an IP address:

```
crypto pki trustpoint local
 enrollment selfsigned
 end
configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
crypto pki enroll local
Nov 29 20:51:13.067: %SSH-5-ENABLED: SSH 1.99 has been enabled
Nov 29 20:51:13.267: %CRYPTO-6-AUTOGEN: Generated new 512 bit key pair
% Include the router serial number in the subject name? [yes/no]: yes
% Include an IP address in the subject name? [no]: yes
Enter Interface name or IP Address[]: ethernet 0
Generate Self Signed Router Certificate? [yes/no]: yes
Router Self Signed Certificate successfully created
```

Note A router can have only one self-signed certificate. If you attempt to enroll a trustpoint configured for a self-signed certificate and one already exists, you receive a notification and are asked if you want to replace it. If so, a new self-signed certificate is generated to replace the existing one.

Enabling the HTTPS Server Example

The following example shows how to enable the HTTPS server and generate a default trustpoint because one was not previously configured:

```
configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ip http secure-server
% Generating 1024 bit RSA keys ...[OK]
*Dec 21 19:14:15.421:%PKI-4-NOAUTOSAVE:Configuration was modified.  Issue "write memory"
to save new certificate
Router(config)#
```

Note You need to save the configuration to NVRAM if you want to keep the self-signed certificate and have the HTTPS server enabled following router reloads.

The following message also appears:

```
*Dec 21 19:14:10.441:%SSH-5-ENABLED:SSH 1.99 has been enabled
```

Note Creation of the key pair used with the self-signed certificate causes the Secure Shell (SSH) server to start. This behavior cannot be suppressed. You may want to modify your Access Control Lists (ACLs) to permit or deny SSH access to the router. You can use the **ip ssh rsa keypair-name** *unexisting-key-pair-name* command to disable the SSH server.

Verifying the Self-Signed Certificate Configuration Example

The following example displays information about the self-signed certificate that you just created:

```
Router# show crypto pki certificates
Router Self-Signed Certificate
  Status: Available
  Certificate Serial Number: 01
  Certificate Usage: General Purpose
  Issuer:
    cn=IOS-Self-Signed-Certificate-3326000105
  Subject:
    Name: IOS-Self-Signed-Certificate-3326000105
    cn=IOS-Self-Signed-Certificate-3326000105
  Validity Date:
    start date: 19:14:14 GMT Dec 21 2004
    end   date: 00:00:00 GMT Jan 1 2020
  Associated Trustpoints: TP-self-signed-3326000105
```

Note The number 3326000105 is the router’s serial number and varies depending on the router’s actual serial number.

The following example displays information about the key pair corresponding to the self-signed certificate:

```
Router# show crypto key mypubkey rsa
% Key pair was generated at: 19:14:10 GMT Dec 21 2004
Key name: TP-self-signed-3326000105
 Usage: General Purpose Key
 Key is not exportable.
 Key Data:
  30819F30 0D06092A 864886F7 0D010101 05000381 8D003081 89028181 00B88F70
  6BC78B6D 67D6CFF3 135C1D91 8F360292 CA44A032 5AC1A8FD 095E4865 F8C95A2B
  BFD1C2B7 E64A3804 9BBD7326 207BD456 19BAB78B D075E78E 00D2560C B09289AE
  6DECB8B0 6672FB3A 5CDAEE92 9D4C4F71 F3BCB269 214F6293 4BA8FABF 9486BCFC
  2B941BCA 550999A7 2EFE12A5 6B7B669A 2D88AB77 39B38E0E AA23CB8C B7020301 0001
% Key pair was generated at: 19:14:13 GMT Dec 21 2004
Key name: TP-self-signed-3326000105.server
 Usage: Encryption Key
 Key is not exportable.
 Key Data:
  307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00C5680E 89777B42
  463E5783 FE96EA9E F446DC7B 70499AF3 EA266651 56EE29F4 5B003D93 2FC9F81D
  8A46E12F 3FBAC2F3 046ED9DD C5F27C20 1BBA6B9B 08F16E45 C34D6337 F863D605
  34E30F0E B4921BC5 DAC9EBBA 50C54AA0 BF551BDD 88453F50 61020301 0001
```

Note The second key pair with the name TP-self-signed-3326000105.server is the SSH key pair and is generated when any key pair is created on the router and SSH starts up.

The following example displays information about the trustpoint named “local”:

```
Router# show crypto pki trustpoints
Trustpoint local:
    Subject Name:
    serialNumber=C63EBBE9+ipaddress=10.3.0.18+hostname=test.example.com
          Serial Number: 01
    Persistent self-signed certificate trust point
```

Configuring Direct HTTP Enrollment Example

The following example show how to configure an enrollment profile for direct HTTP enrollment with a CA server:

```
crypto pki trustpoint Entrust
 enrollment profile E
 serial
crypto pki profile enrollment E
 authentication url http://entrust:81
 authentication command GET /certs/cacert.der
 enrollment url http://entrust:81/cda-cgi/clientcgi.exe
 enrollment command POST reference_number=$P2&authcode=$P1
 &retrievedAs=rawDER&action=getServerCert&pkcs10Request=$REQ
 parameter 1 value aaaa-bbbb-cccc
 parameter 2 value 5001
```

Configuring Certificate Enrollment in a Two-Tier PKI Environment Example

Example of importing the ROOT-CA via terminal.

```
(config)#crypto pki trustpoint ROOT-CA
(ca-trustpoint)#revocation-check none
(ca-trustpoint)#enrollment terminal


(config)#crypto pki authenticate ROOT-CA

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

-----BEGIN CERTIFICATE-----
MIIDdTCCAl2gAwIBAgIQIfTArEE1yKZPXHaAVgDk5jANBgkqhkiG9w0BAQsFADBN
MRMwEQYKCZImiZPyLGQBGRYDY29tMRgwFgYKCZImiZPyLGQBGRYIdnBuLWVhc3Qx
HDAaBgNVBAMTE3Zwbi1lYXN0LXphY2ttY2ktQ0EwHhcNMTgxMjIwMDAwNjMyWhcN
MjgxMjIwMDAxNjMyWjBNMRMwEQYKCZImiZPyLGQBGRYDY29tMRgwFgYKCZImiZPy
LGQBGRYIdnBuLWVhc3QxHDAaBgNVBAMTE3Zwbi1lYXN0LXphY2ttY2ktQ0EwggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQC9Gdns9lU2HHc+XYhrmZKg6+Xo
5kNflu6mMgCfZ7ZiAKxZ03whJWZqNC7JRZQ+LkIJAcBUSf2mSJWRp+HVgI6k4Zf7
bMgIBq629HT8XmFLrr3lfh1lfL7WqI1Uez7/PEzjsw09y/m/WiSnrlgR3+PvyDbH
E86A6JnmtTNIs4qawUe72BlnEzwwRaFNi7VQz7GQw3CUo+RX9wtFYjABTyTUM/BA
MP47pI8CVh1jHVHqHcbqpyd97j1/8n1d/NCmcHKIg2hnKEO1Hx8oK7QIHe1rkryl
+r0ol2fS3CGgY000+FINs3qw4h8H8xfmsc5cs8lJCIbZGJhMTXq6u4Ecp+N1AgMB
AAGjUTBPMAsGA1UdDwQEAwIBhjAPBgNVHRMBAf8EBTADAQH/MB0GA1UdDgQWBBTb
zvfa7aNZspz3GwJCvKDIKO8KFTAQBgkrBgEEAYI3FQEEAwIBADANBgkqhkiG9w0B
AQsFAAOCAQEAgTIPTauHsPp7h1v/iFXkbVV1aG7O8/IaJG0sCr0f9/nsfM9HO0Jm
LP+twy5KkFa7I6u4vMlMlfNyujS60Fqnw3m8UJCy2SkYVwlGrBddN+BQbnkZ460M
sYfaynFBsvsbmmaLEqUQ3t9cmNCskXoda+FffyFTwAUBFzV66BGKpn6Y7oyIghF5
NLjjgWPVmRy7RKM4IKe9J0+oEmnugwtdfHgiFdX+d6qPovjbApj2j6N4+Cv6qHDO
/c+wUXRxz08eFNOqHNJipk70OXMrUh4UaWMnM/CYA9E1sjjSAWhBl4ii/+fiaILw
xgof+2mmIzafzFZz+eVf5kgwpV07GlZlng==
-----END CERTIFICATE-----
quit
Certificate has the following attributes:
       Fingerprint MD5: 99182E1E 96FB0595 DF86BFCE 3C781CF5
      Fingerprint SHA1: 6E55B878 9AA3B603 D689AC25 F027615E 0C88E6E4

% Do you accept this certificate? [yes/no]: yes

Authenticating SUB-CA without having to specify or accept the fingerprint.

(config)#crypto pki trustpoint SUB-CA
(ca-trustpoint)#enrollment url  http://<SUBCA_IP/FQDN>:80/certsrv/mscep/mscep.dll
(ca-trustpoint)#chain-validation continue ROOT-CA
(ca-trustpoint)#revocation-check none

(ca-trustpoint)#crypto pki authenticate SUB-CA
Certificate has the following attributes:
       Fingerprint MD5: 5C38CB0A 050AAE87 84A08A75 5F7084B8
      Fingerprint SHA1: EB829470 B8B9E26E 4457F346 7A3E957C C623C6F9
Certificate validated - Signed by existing trustpoint CA certificate.

Trustpoint CA certificate accepted.
```

Additional References

Related Documents

Related Topic	Document Title
USB token RSA operations: Benefits of using USB tokens	"Storing PKI Credentials" module in the Cisco IOS Security Configuration Guide: Secure Connectivity
USB token RSA operations: Certificate server configuration	"Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment" chapter in the Cisco IOS Security Configuration Guide: Secure Connectivity See the "Generating a Certificate Server RSA Key Pair" section, the "Configuring a Certificate Server Trustpoint" section, and related examples.
Overview of PKI, including RSA keys, certificate enrollment, and CAs	" Cisco IOS PKI Overview: Understanding and Planning a PKI " module in the Cisco IOS Security Configuration Guide: Secure Connectivity
Secure Device Provisioning: functionality overview and configuration tasks	" Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI " module in the Cisco IOS Security Configuration Guide: Secure Connectivity
RSA key generation and deployment	" Deploying RSA Keys Within a PKI " module in the Cisco IOS Security Configuration Guide: Secure Connectivity
Cisco IOS certificate server overview information and configuration tasks	" Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment " module in the Cisco IOS Security Configuration Guide: Secure Connectivity
Setting up and using a USB token	" Storing PKI Credentials " module in the Cisco IOS Security Configuration Guide: Secure Connectivity
Cisco IOS security commands	*Cisco IOS Security Command Reference*
Suite-B ESP transforms	Configuring Security for VPNs with IPsec feature module.
Suite-B SHA-2 family (HMAC variant) and Elliptic Curve (EC) key pair configuration.	Configuring Internet Key Exchange for IPsec VPNs feature module.
Suite-B Integrity algorithm type transform configuration.	Configuring Internet Key Exchange Version 2 (IKEv2) feature module.
Suite-B Elliptic Curve Digital Signature Algorithm (ECDSA) signature (ECDSA-sig) authentication method configuration for IKEv2.	Configuring Internet Key Exchange Version 2 (IKEv2) feature module.

Related Topic	Document Title
Suite-B Elliptic curve Diffie-Hellman (ECDH) support for IPsec SA negotiation	Configuring Internet Key Exchange for IPsec VPNs and Configuring Internet Key Exchange Version 2 (IKEv2) feature modules.
Recommended cryptographic algorithms	*Next Generation Encryption*

MIBs

MIB	MIBs Link
None.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for PKI Certificate Enrollment

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

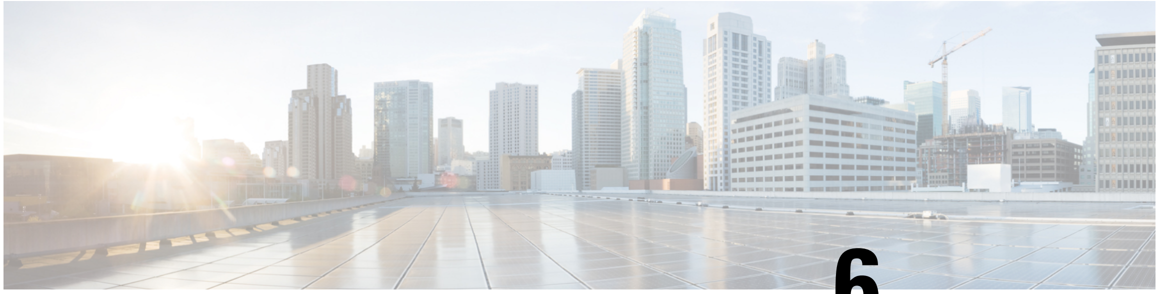
Table 4: Feature Information for PKI Certificate Enrollment

Feature Name	Releases	Feature Information
Certificate Autoenrollment		This feature introduces certificate autoenrollment, which allows the router to automatically request a certificate from the CA that is using the parameters in the configuration. The following commands were introduced by this feature: **auto-enroll**, **rsakeypair**, **show crypto ca timers**.
Certificate Enrollment Enhancements		This feature introduces five new **crypto ca trustpoint**commands that provide new options for certificate requests and allow users to specify fields in the configuration instead of having to go through prompts. The following commands were introduced by this feature: **ip-address**(ca-trustpoint), **password**(ca-trustpoint), **serial-number**, **subject-name**, **usage**.
Direct HTTP Enrollment with CA Servers		This feature allows users to configure an enrollment profile if their CA server does not support SCEP and they do not want to use an RA-mode CS. The enrollment profile allows users to send HTTP requests directly to the CA server instead of to an RA-mode CS. The following commands were introduced by this feature: **authentication command**, **authentication terminal**, **authentication url**, **crypto ca profile enrollment**, **enrollment command**, **enrollment profile**, **enrollment terminal**, **enrollment url**, **parameter**.

Feature Name	Releases	Feature Information
Import of RSA Key Pair and Certificates in PEM Format		This feature allows customers to issue certificate requests and receive issued certificates in PEM-formatted files. The following commands were modified by this feature: **enrollment**, **enrollment terminal**.
Key Rollover for Certificate Renewal		This feature allows the certificate renewal request to be made before the certificate expires and retains the old key and certificate until the new certificate is available. The following commands were introduced or modified by this feature: **auto-enroll**, **regenerate**.
Manual Certificate Enrollment (TFTP Cut-and-Paste)		This feature allows users to generate a certificate request and accept CA certificates and the router's certificates via a TFTP server or manual cut-and-paste operations. The following commands were introduced or modified by this feature: **crypto ca import**, **enrollment**, **enrollment terminal**.
Persistent Self-Signed Certificates		This feature allows the HTTPS server to generate and save a self-signed certificate in the router startup configuration. Thus, future SSL handshakes between the client and the HTTPS server can use the same self-signed certificate without user intervention. The following commands were introduced or modified by this feature: **enrollment selfsigned**, **show crypto pki certificates**, **show crypto pki trustpoints**.

Feature Name	Releases	Feature Information
PKI Status		This enhancement adds the **status** keyword to the **show crypto pki trustpoints** command, which allows you to display the current status of the trustpoint. **Note** This is a minor enhancement. Minor enhancements are not typically listed in Feature Navigator.
Reenroll Using Existing Certificates		This feature allows users to reenroll a router with a Cisco IOS CA via existing certificates from a third-party vendor CA. The following commands were introduced by this feature: **enrollment credential**, **grant auto trustpoint**.

Feature Name	Releases	Feature Information
Suite-B support in IOS SW crypto		Suite-B adds the following support for certificate enrollment for a PKI: • Elliptic Curve Digital Signature Algorithm (ECDSA) (256 bit and 384 bit curves) is used for the signature operation within X.509 certificates. • PKI support for validation of for X.509 certificates using ECDSA signatures. • PKI support for generating certificate requests using ECDSA signatures and for importing the issued certificates into IOS. Suite-B requirements comprise of four user interface suites of cryptographic algorithms for use with IKE and IPsec that are described in RFC 4869. Each suite consists of an encryption algorithm, a digital signature algorithm, a key agreement algorithm, and a hash or message digest algorithm. See the *Configuring Security for VPNs with IPsec* feature module for more detailed information about Cisco IOS Suite-B support.
Trustpoint CLI		This feature introduces the **crypto pki trustpoint** command, which adds support for trustpoint CAs.



CHAPTER 6

Setting Up Secure Device Provisioning for Enrollment in a PKI

This module describes how to use Secure Device Provisioning (SDP) in a public key infrastructure (PKI). SDP is a web-based certificate enrollment interface that can be used to easily deploy PKI between two end devices, such as a Cisco IOS client and a Cisco IOS certificate server. The end devices may or may not be directly connected to the network at the time of deployment or provisioning. SDP provides a solution for users deploying a large number of peer devices (including certificates and configurations).

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the Next Generation Encryption (NGE) white paper.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI

Setting Up SDP for Enrollment in a PKI

Before you set up SDP, your environment should meet the following requirements:

- The petitioner device and the server must have IP connectivity between each other.
- The introducer must have a web browser that supports JavaScript.
- The introducer must have enable privileges on the client device.
- A Cisco IOS Release 12.3(8)T PKI-enabled image or a later image.

Setting Up SDP for Enrollment in a PKI Using USB Tokens

To leverage USB tokens to provision devices with SDP, your environment should meet the following requirements:

- Both the petitioner device and the server must have IP connectivity between each other.
- The introducer must have a web browser that supports JavaScript.
- The introducer must have enable privileges on the client device.
- The introducer must have access to a petitioner device.
- The introducer must have access to the USB token and PIN, if configured.
- A Cisco IOS Release 12.4(15)T PKI-enabled image or a later image.

Note Cisco IOS Release 12.4(15)T or a later release provides the flexibility to move credentials stored on the USB token. However, the device used to configure the USB token may run any Cisco IOS Release 12.3(14)T PKI-enabled image or a later image.

Using SDP to Configure a Device for an Internet Connection Through a Service Provider

To leverage SDP to configure a device that is not connected to the Internet, your environment should meet the following requirements:

- The introducer must have a web browser that supports JavaScript.
- The introducer must have enable privileges on the client device.
- A Cisco router that supports a DHCP client and a PPPoE client and has a configured LAN or WAN interface.
- A Cisco IOS Release 12.4(20)T PKI-enabled image or a later image. If a previous Cisco IOS release is used on one of the devices, the SDP functionality defaults to the earlier Cisco IOS version.

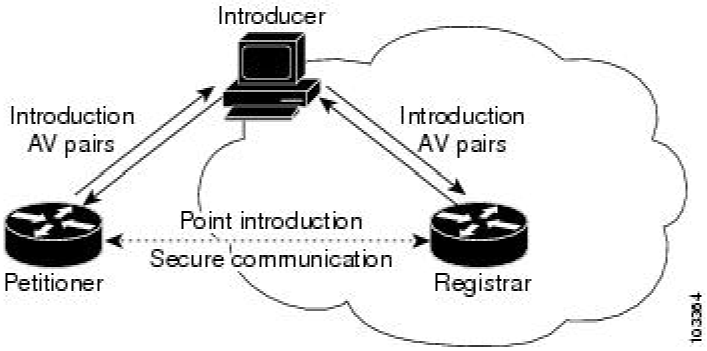
Information About Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI

SDP Overview

SDP (also refer red to as Trusted Transitive Introduction [TTI]) is a communication protocol that provides a bidirectional introduction between two end entities, such as a new network device and a Virtual Private Network (VPN). SDP involves the following three entities (see the figure below):

- Introducer--A mutually trusted device that introduces the petitioner to the registrar. The introducer can be a device user, such as a system administrator.
    - An introducer can be configured as an administrative introducer, which allows an administrator performing the introduction to supply the name for the device being introduced. The supplied device name is used as if it were the name of an introducer in the normal SDP mechanisms, preserving the existing functionality of the SDP configuration. For more information on function of the administrative introducer, see the section Authentication and Authorization Lists for an Administrative Introducer, on page 134.
- Petitioner--A client, or new device, to be introduced to the secure network.
- Registrar--A server that authorizes the petitioner. The registrar can be a certificate server.

Figure 4: Post-Introduction Secure Communication



As of Cisco IOS Release 12.4(20)T or a later release, the introducer can start the SDP process without establishing prior Internet connectivity on the petitioner. The use of the prep-connect phase and the connect phase provides the ability to configure a petitioner for Internet connectivity through a service provider. See the How SDP Works, on page 124 for more information on the prep-connect phase and the connect phase.

The registrar communicates directly with an external authentication, authorization, and accounting (AAA) server to verify petitioner credentials, permit or deny enrollment, and retrieve specific petitioner configuration information. The petitioner and registrar serve web pages to the introducer, the end user. The petitioner receives the bootstrap configuration from a remote management system through the introducer’s web browser.

SDP is implemented over a web browser with six possible phases--prep-connect (optional), connect, start (optional), welcome, introduction, and completion. Each phase is shown to the user through a web page. See the How SDP Works, on page 124 for more information on each phase.

How SDP Works

The following sections describe how SDP deploys PKI between two devices:

- SDP Prep-Connect Phase, on page 124
- SDP Connect Phase, on page 125
- SDP Start Phase, on page 127
- SDP Welcome Phase, on page 128
- SDP Introduction Phase, on page 129
- SDP Completion Phase, on page 130

The SDP process starts with one of three entry pages being loaded into the web browser by the introducer: the SDP prep-connect phase received from the administrator; the start phase loaded from the registrar; or the welcome phase loaded from the petitioner.

The sample figures show how to introduce the local device (the petitioner) to the secure domain of the registrar. The "introducer" is referred to as the end user.

SDP Prep-Connect Phase

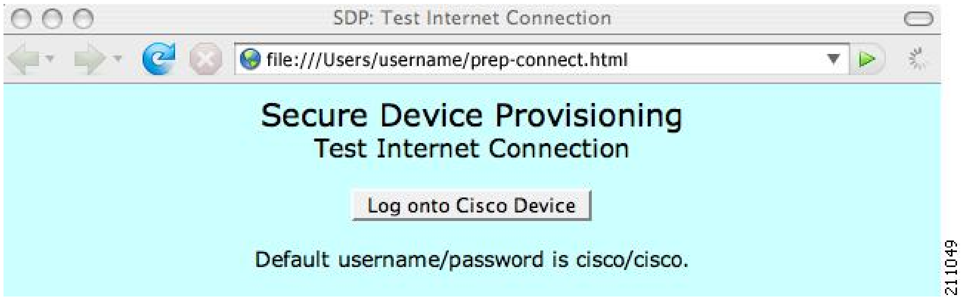
The prep-connect page is optional. Without the prep-connect page, the petitioner must have IP connectivity established.

The administrator must configure the prep-connect template and send the prep-connect page to the introducer. See the Default Prep-Connect Template, on page 139 for more information.

The administrator must also obtain and communicate the username and password for the secure network to the introducer by a telephone call, an e-mail, a secure e-mail, a CD, or a USB token. The registrar may be configured to authenticate the introducer using an existing AAA infrastructure (for example, an existing username and password database that is part of the existing corporate domain). The SDP prep-connect phase supports a challenge password mechanism as is used by common AAA infrastructures. See the How SDP Uses an External AAA Database, on page 133 for more information.

After receiving the prep-connect page, the introducer must load the page onto the computer where the HTTP browser operates. The introducer then loads the prep-connect page into the HTTP browser as a local file and then the prep-connect page is displayed (see the figure below).

Figure 5: Sample SDP Prep-Connect Page



SDP: Test Internet Connection

file:///Users/username/prep-connect.html

Secure Device Provisioning

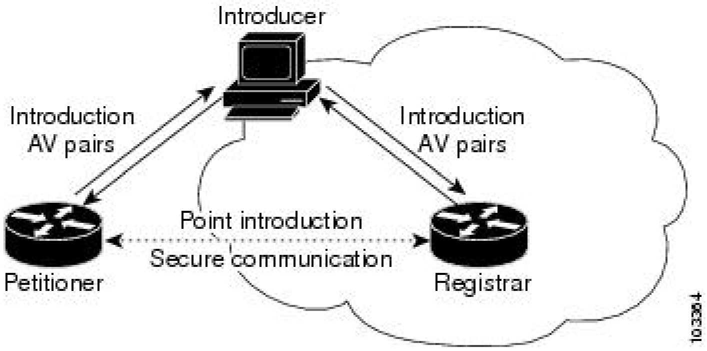
Test Internet Connection

Log onto Cisco Device

Default username/password is cisco/cisco.

After the introducer clicks the Log onto Cisco Device button, the login dialog box is displayed (see the figure below). The introducer enters the factory default username (cisco) and password (cisco) of the Cisco device.

Figure 6: Sample Petitioner Login Dialog Box



The introducer authenticates with the petitioner and then Internet connectivity is tested by attempting to access a known URL. Access to www.cisco.com (198.133.219.25) is tested by default. The administrator can modify the URL to be used for testing connectivity by modifying the default prep-connect template. For more information about modifying the default test URL and other fields that the administrator may configure for the prep-connect page, see the section Default Prep-Connect Template, on page 139.

Note To mitigate the possibility that the prep-connect page could be modified to contain an IP address of an untrusted registrar or that a prep-connect page might be e-mailed from an untrusted source, use a secure method, such as secure e-mail, to send the prep-connect page.

If Internet connectivity is established either the start page or welcome page is displayed, depending on the prep-connect template setting as defined by the administrator. If Internet connectivity is not established, the connect page is displayed.

SDP Connect Phase

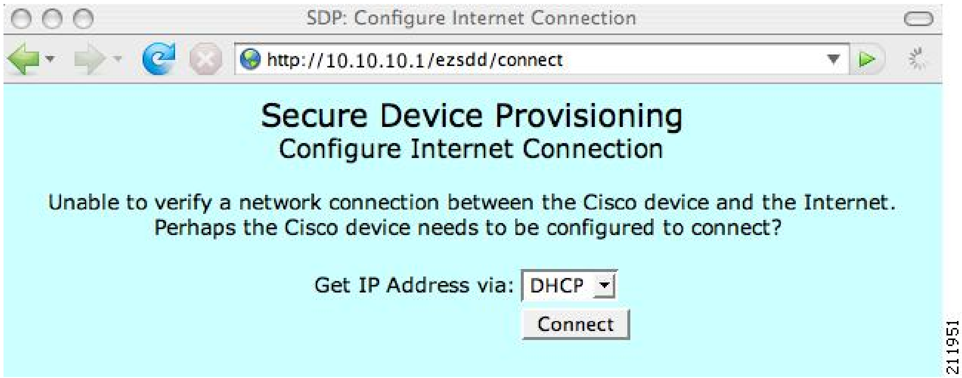
The connect page is displayed only if the prep-connect page is used and there is no IP connectivity for the petitioner at the completion of the prep-connect phase. The connect page has three IP address assignment methods to allow flexibility for your Cisco IOS platform: Dynamic Host Configuration Protocol (DHCP), Point to Point Protocol over Ethernet (PPPoE), or static IP address assignment.

Note SDP functionality is not used with the Cisco IOS configuration to establish Internet connectivity. SDP functionality includes a signature on the Cisco IOS configuration, guaranteeing that the values have not changed in transit.

DHCP IP Address Assignment Method

If the introducer chooses DHCP, the default method, for the IP address assignment method option (see the figure below), clicking the Connect button causes the petitioner to be configured for Internet connectivity.

Figure 7: Sample Connect Page for DHCP IP Address Assignment Method



SDP: Configure Internet Connection

http://10.10.10.1/ezsdd/connect

Secure Device Provisioning
Configure Internet Connection

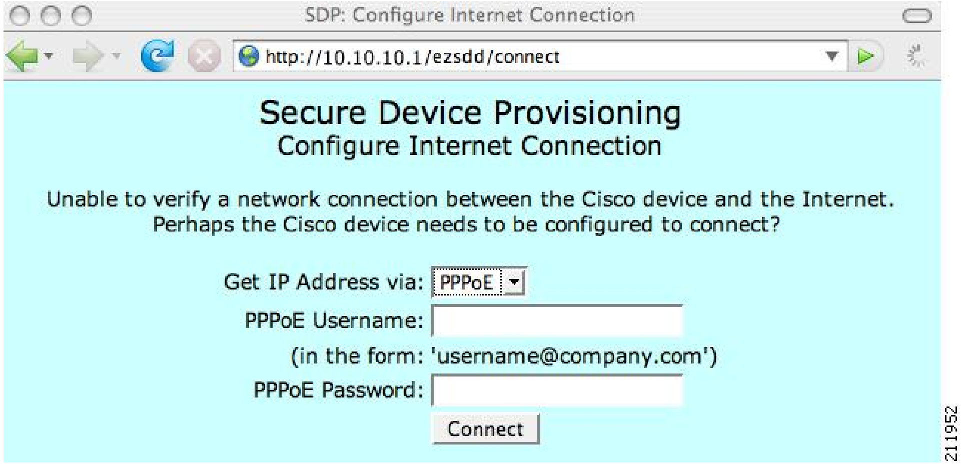
Unable to verify a network connection between the Cisco device and the Internet.
Perhaps the Cisco device needs to be configured to connect?

Get IP Address via: DHCP

PPPoE IP Address Assignment Method

If the introducer chooses PPPoE, input fields for PPPoE username and password are displayed (see the figure below). The introducer must enter the username and password as supplied by the Internet service provider (ISP) and then click the Connect button, which causes petitioner to be configured for Internet connectivity.

Figure 8: Sample Connect Page for PPPoE IP Address Assignment Method



SDP: Configure Internet Connection

http://10.10.10.1/ezsdd/connect

Secure Device Provisioning
Configure Internet Connection

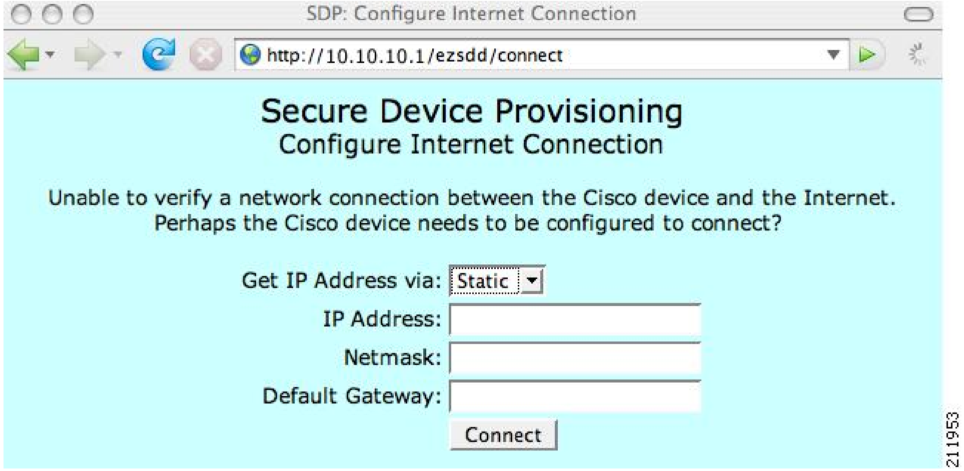
Unable to verify a network connection between the Cisco device and the Internet.
Perhaps the Cisco device needs to be configured to connect?

Get IP Address via: PPPoE
PPPoE Username:
(in the form: 'username@company.com')
PPPoE Password:

Static IP Address Assignment Method

If the introducer chooses static, input fields for the IP address, netmask, and the default gateway are displayed (see the figure below). The introducer must enter the configuration values as supplied by the ISP and then click the Connect button, which causes petitioner to be configured for Internet connectivity.

Figure 9: Connect Page for Static IP Address Assignment Method



SDP: Configure Internet Connection

http://10.10.10.1/ezsdd/connect

Secure Device Provisioning
Configure Internet Connection

Unable to verify a network connection between the Cisco device and the Internet.
Perhaps the Cisco device needs to be configured to connect?

Get IP Address via: Static
IP Address:
Netmask:
Default Gateway:

Connect Page IP Address Configuration

After IP address configuration, Internet connectivity is tested again by attempting to access a known URL configured by the administrator in the prep-connect template (www.cisco.com by default). If Internet connectivity is now established either the start page or welcome page is displayed, depending on the prep-connect template setting as defined by the administrator. If Internet connectivity is not established, the introducer should verify the settings entered or contact their administrator.

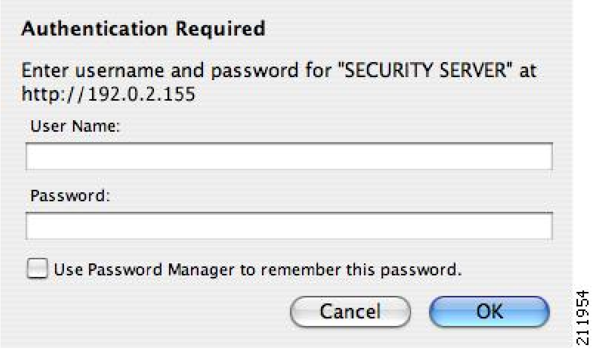
SDP Start Phase

The start page is optional. Without the start page, during the SDP exchange, the user clicks the Next button on the welcome page and is sent to the registrar's introduction page. Because the user has not previously connected to the registrar, he or she is required to log in to the registrar using available credentials (per the registrar configuration). Some browsers fail to reconnect to the registrar after the user has entered the login data. As of Cisco IOS Release 12.4(4)T, users may configure their browsers to begin the SDP exchange by contacting the registrar's introduction URL through a start page. Thereafter, the registrar can direct the user to the welcome page, which is on the petitioner device. The SDP transaction continues through the welcome, introduction, and completion phases as described in this document.

To begin the SDP transaction from the registrar, the user must configure the browser through the **template http start** command; otherwise, the SDP transaction must begin from the welcome page on the petitioner. See the How Custom Templates Work with SDP, on page 134.

Before the welcome page is displayed, the user must direct his or her browser to the start page through the URL http://registrar/ezsdd/intro. A login dialog box is then displayed, and the end user can log into the registrar through a username and password supplied by the administrator to access the secure network (see the figure below).

Figure 10: Registrar Remote Login Dialog Box



Authentication Required

Enter username and password for "SECURITY SERVER" at http://192.0.2.155

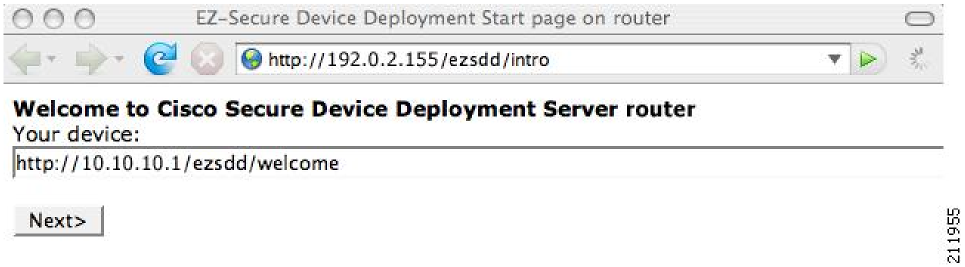
User Name:

Password:

□ Use Password Manager to remember this password.

After entering a valid username and password, the start page is displayed (see the figure below).

Figure 11: Sample SDP Start Page



EZ-Secure Device Deployment Start page on router

http://192.0.2.155/ezsdd/intro

Welcome to Cisco Secure Device Deployment Server router

Your device:

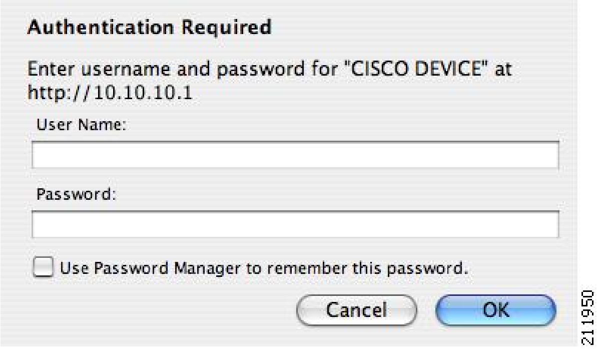
http://10.10.10.1/ezsdd/welcome

The user must log into the petitioner through the URL http://10.10.10.1/ezsdd/welcome. The welcome phase begins when the user clicks the Next button on the start page.

SDP Welcome Phase

The local login dialog box is then displayed (see the figure below), and the end user can log into the local device through the factory default username (cisco) and password (cisco). The welcome page is then displayed.

Figure 12: Petitioner Local Login Dialog Box



Authentication Required

Enter username and password for "CISCO DEVICE" at http://10.10.10.1

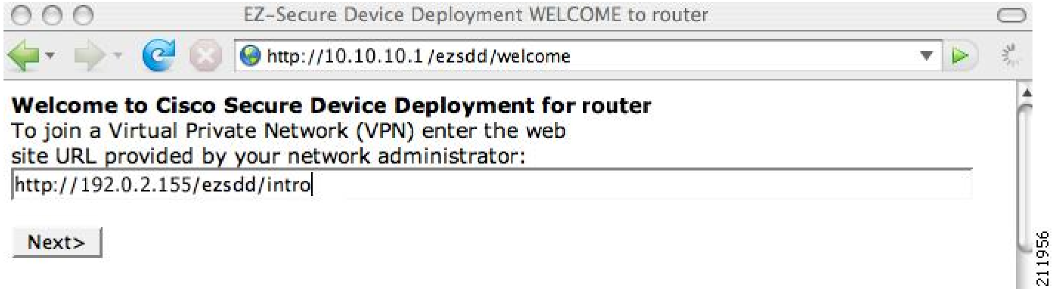
User Name:

Password:

□ Use Password Manager to remember this password.

After the password is successfully entered, the welcome web page is displayed (see the figure below), which is served by the petitioner.

Figure 13: Sample SDP Welcome Page



EZ-Secure Device Deployment WELCOME to router

http://10.10.10.1/ezsdd/welcome

Welcome to Cisco Secure Device Deployment for router
To join a Virtual Private Network (VPN) enter the web
site URL provided by your network administrator:

http://192.0.2.155/ezsdd/intro

Next>

After entering the URL of the registrar (for example, http://192.0.2.155/ezsdd/intro) and clicking the Next button on the welcome web page, the SDP introduction phase begins and the introduction page, which is served by the registrar, is displayed.

SDP Introduction Phase

Before the introduction page is displayed, the end user must log into the registrar if the user has not already done so from the start page (see “SDP Start Phase, on page 127”), which utilizes the external AAA database.

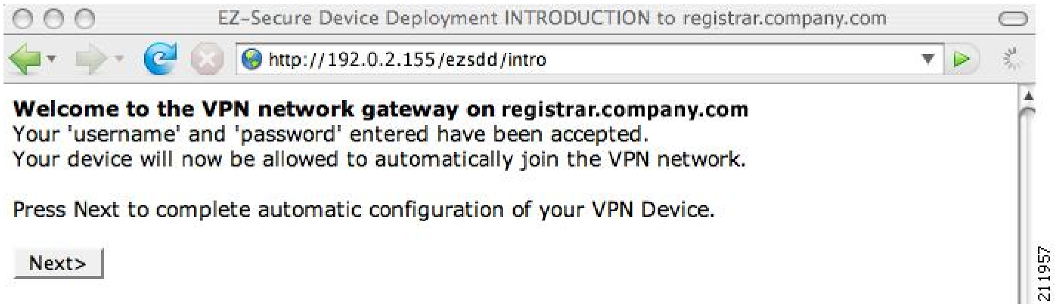
With an external AAA database, the introducer can use an account on the database to perform the introduction without requiring knowledge of the enable password of the registrar. Without an external AAA database, the introducer may use the enable password of the registrar for authentication.

Note Using the enable password of the registrar exposes the password to end users; therefore, it is recommended that the enable password be used for administrative testing only.

The administrative introducer is identified by the HTTP authentication for the introduction page (or the start page), with the AAA database query returning administrative privilege for the user. If the introducer has administrator privilege, the device name is that which was entered in the administrative introduction page. If the introducer does not have administrative privileges, the device name is the introducer name. The existing device certificate is the current certificate on the petitioner, which may be the manufacturing identification certificate (MIC). This certificate may or may not exist. For more information on the function of the external AAA database, see the section “How SDP Uses an External AAA Database, on page 133.”

After the end user successfully enters his or her password, the introduction web page is displayed (see the figure below).

Figure 14: Sample SDP Introduction Page



EZ-Secure Device Deployment INTRODUCTION to registrar.company.com

http://192.0.2.155/ezsdd/intro

Welcome to the VPN network gateway on registrar.company.com
Your 'username' and 'password' entered have been accepted.
Your device will now be allowed to automatically join the VPN network.

Press Next to complete automatic configuration of your VPN Device.

Next>

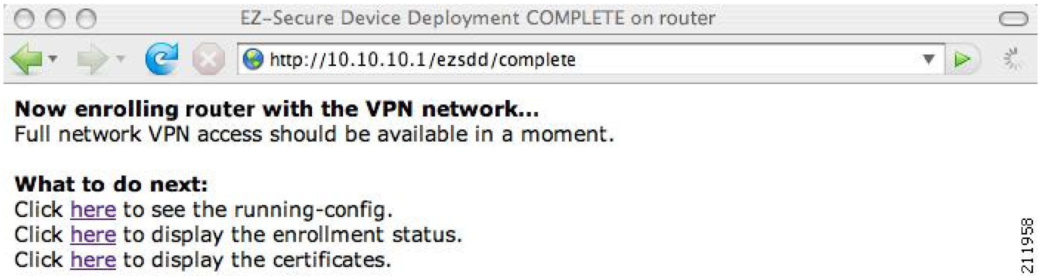
At this point, the registrar passes device information to the external management system to obtain a bootstrap configuration file. For more information on options available to identify a customized bootstrap configuration file, see the section Custom HTML Template Expansion Rules, on page 135.

After the end user clicks the Next button on the introduction page, the end user enters the completion phase and automatically returns to his or her local device.

SDP Completion Phase

Now that the end user has enrolled the petitioner with the registrar, the petitioner serves the completion page (see the figure below).

Figure 15: Sample SDP Completion Page



EZ-Secure Device Deployment COMPLETE on router
http://10.10.10.1/ezsdd/complete
211958

Now enrolling router with the VPN network...
Full network VPN access should be available in a moment.

What to do next:
Click here to see the running-config.
Click here to display the enrollment status.
Click here to display the certificates.

The SDP exchange is now complete. The petitioner has received configuration information from the registrar and should receive a certificate from the registrar shortly.

SDP Leveraging USB Tokens

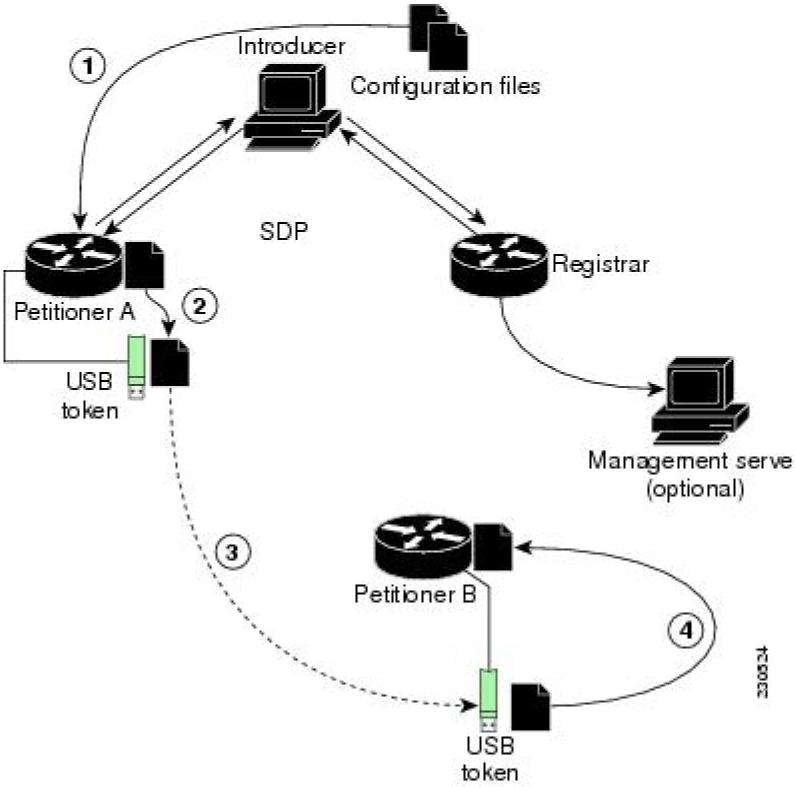
SDP provides for highly scalable deployments and streamlines the deployment of an individual device or multiple devices. USB tokens provide for secure storage and configuration distribution.

As of Cisco IOS Release 12.4(15)T or a later release, USB tokens may be utilized to transfer PKI credentials using SDP to a remote device, and SDP may be used to configure the USB token. The USB token may then be used to provision a device at the same location, or the USB token may be transported to another location where it may be used to provision a remote device.

An example SDP deployment using a USB token to transfer PKI credentials is shown in the figure below. The required devices include the USB token and the SDP entities required to provision a device. These SDP entities are the introducer, the registrar, a petitioner at the local location, Petitioner A, and a petitioner at the remote location, Petitioner B. Optionally, a management server may be used.

Note An optional configuration would be to configure one device as both the registrar and a petitioner, which may be beneficial when the USB token is transported to a remote location. The remote location would not require a separate petitioner device.

Figure 16: Example SDP Environment Using USB Tokens to Transfer Credentials



Use of SDP to Configure the USB Token

Prior to initiating an SDP introduction a USB token is inserted into the petitioner device. In the example configuration shown in the figure, the USB token would be inserted into Petitioner A. The petitioner may be configured to ignore any existing information on the USB token. As in regular SDP operations, for a scalable configuration of USB tokens, an initial template configuration has to be prepared and placed onto each SDP device with appropriate target configuration information.

Files used to provision a device are moved in the following sequence.

1. One petitioner, Petitioner A, is at the local location. petitioner A engages directly with the SDP exchange to perform the initial configuration of the USB token. Files used to configure the USB token, binary files and template files, are retrieved from the registrar and moved to Petitioner A.

The URL for the binary file location is expanded on the registrar. Binary files are not processed through the template expansion functions. The template expansion occurs on the registrar for both the source URL and destination URL.

By default, binary files and template files are retrieved from and stored to NVRAM on the registrar and petitioner respectively. The binary file location on the registrar and the destination binary file location on Petitioner A may be specified with the **binary file**command. The template file location on the registrar and the destination template file location on Petitioner A may be specified with the **template file**command.

1. The Rivest, Shamir, and Adelman (RSA) keys and certificate chain information are moved from Petitioner A to the USB token.

2. The USB token is transported to the remote location where it is inserted into Petitioner B.
3. The configuration files on the USB token are used to provision the local device. Files from the USB token may be moved to a storage location on Petitioner B with the **crypto key move rsa** command.

SDP Phases with a USB Token

The same SDP phase concepts introduced in the "SDP Overview" section are used, with the following distinctions in the SDP welcome phase, the SDP introduction phase, and the SDP completion phase.

SDP Welcome Phase with a USB Token

The SDP welcome phase begins as usual, when an introduction is initiated by connecting to the welcome user interface. If there is an existing certificate on the USB token, it is used for signing the SDP exchange. Instead of a local RSA key pair, a new RSA key pair on the token is used.

Note The RSA key pair generation may take a substantial length of time, anywhere from 5 to 10 minutes if the key is generated on the token. The length of time is dependent on hardware key generation routines available on the USB token. An informative web page is presented to the introducer, indicating that RSA key pair generation is occurring.

The new key pair generated by Petitioner A is added to the USB token without removing any existing RSA key pairs. SDP AV pairs indicate both that a token is being used and if there is any token secondary configuration information. If an optional management server is in use, the AV pair information is used to determine if any special configuration commands are needed.

SDP Introduction Phase with a USB Token

The SDP Introduction phase begins with AV pairs being transferred to the registrar. When the registrar detects USB token related AV pairs, the registrar, if previously configured, may prepare configuration information destined for the USB token. Currently configuration commands are sent as a specific configuration files that are subsequently merged with the running configuration.

The administrator can leverage normal SDP configuration commands to configure the USB token. USB token information that should be configured includes the certificate, the bootstrap configuration, and the PIN number configuration.

SDP Completion Phase with a USB Token

At the beginning of the completion phase, the introduction proceeds with AV pairs being transferred to the petitioner. The various files are stored in the specified file system locations and then the existing configuration file processing proceeds. This ordering allows the configuration to take advantage of the new files that have been transferred.

Use of the Configured USB Token

After the USB token is configured by Petitioner A, it is transported from its current location to the remote location, where the second petitioner, Petitioner B is located. The USB token is inserted into the target device, Petitioner B, which then inherits the USB token configuration and cryptographic material from the USB token. The end user at the remote location must have the PIN number on the USB token. The PIN number is either the default factory PIN or the PIN number the administrator configured during the introduction phase.

How SDP Uses an External AAA Database

The external AAA database is accessed twice during the SDP exchange. The first time the AAA database is accessed, the introducer is authenticated; that is, when the registrar receives an introduction request through the secure HTTP (HTTPS) server, the registrar does an AAA lookup based on the introducer's username and password to authorize the request. The second time the AAA database is accessed, authorization information is obtained and applied to the configuration and certificates that are issued to the petitioner device; that is, the registrar checks the integrity of the request by verifying the request signature using the petitioner-signing certificate. The certificate subject name may be specified in the AAA database, and up to nine configuration template variables may be specified and expanded into the template configuration.

Use of a Self-Signed Certificate Versus a Certificate Issued by Another CA Server

By default, the SDP exchange results in only one certificate being issued to the petitioner device. Although just one certificate is issued, the introducer is not restricted from introducing multiple devices and thus obtaining multiple certificates. By specifying the subject name in the certificate that is issued, you can be assured that all certificates that are issued in this way are associated with the introducer. You can use PKI AAA integration to further restrict the use of these certificates. Additionally, the AAA database can be configured to accept only one authentication and authorization request per user.

Because the petitioner certificate is self-signed, it is just used to convey the public key of the petitioner. No verification or authorization check is performed on the certificate; thus, authorization is per-user based and no per-device information is used.

There are some scenarios when per-device authorization is preferred. Therefore, if the petitioner is able to use certificates issued by other certification authority (CA) servers for SDP transactions, the existing PKI can be used and authorization can be achieved over the certificate attributes.

Configuring the petitioner and the registrar for certificate-based authorization provides authorization of the specific device being deployed. Previously, introducer-to-petitioner device communication was secured only using physical security between the introducer and the petitioner device. SDP certificate-based authorization gives the registrar an opportunity to validate the current device identity before accepting the introduction.

Authentication and Authorization Lists for SDP

When you are configuring your SDP registrar, if you specify an authentication list and an authorization list, the registrar uses the specified lists for all introducer requests. The authentication list is used when authenticating the introducer (the AAA server checks for a valid account by looking at the username and password). The authorization list is used to receive the appropriate authorized fields for the certificate subject name and a list of template variables to be expanded into the Cisco IOS command-line interface (CLI) snippet that is sent back to the petitioner. The authentication and authorization lists are usually point to the same AAA server list, but it is possible to use a different database for authentication and authorization. (Storing files on different databases is not recommended.)

When a petitioner makes an introduction request, multiple queries are sent to the AAA list database on the RADIUS or TACACS+ server. The queries search for entries of the following form:

```
user Password <userpassword>
   cisco-avpair="ttti:subjectname=<<DN subjectname>>"
   cisco-avpair="tti:iosconfig#<<value>>"
   cisco-avpair="tti:iosconfig#<<value>>"
   cisco-avpair="tti:iosconfig#=<<value>>"
```

Note The existence of a valid AAA username record is enough to pass the authentication check. The "cisco-avpair=tti" information is necessary only for the authorization check.

If a subject name was received in the authorization response, the SDP registrar stores it in the enrollment database, and that "subjectname" overrides the subject name that is supplied in the subsequent certificate request (PKCS10) from the petitioner device.

The numbered "tti:iosconfig" values are expanded into the SDP Cisco IOS snippet that is sent to the petitioner. The configurations replace any numbered ($1 through $9) template variable. Because the default Cisco IOS snippet template does not include the variables $1 through $9, these variables are ignored unless you configure an external Cisco IOS snippet template. To specify an external configuration, use the **template config** command.

Note The template configuration location may include a variable "$n," which is expanded to the name with which the user is logged in.

Authentication and Authorization Lists for an Administrative Introducer

The SDP mechanisms assume a permanent relationship between the introducer and the device. As a result, the introducer username is used to define the device name.

In some SDP deployment scenarios, the introducer is an administrator doing the introduction for many devices. However, using the introducer (the administrator) name to define the device name results in multiple devices being incorrectly deployed with the same device name. Instead, an administrative introducer allows the administrator to specify the correct device name during the introduction.

More generally stated, the introducer username is used as the database record locator to determine all other information about the device including the Cisco IOS configuration template, various template variables (pulled from an AAA database and expanded into the template), and the appropriate subject name for PKI certificates issued to the device. For simplicity, this database record locator is called the user/device name.

The administrative introducer provides a device name. In that way, an administrator can provide the appropriate record locator when doing an introduction. For example, if an administrator is trying to introduce a device for username "user1," the administrator introduces the device into the PKI network and provides user1 as the record locator after logging into the registrar using the administrator's own credentials. The record locator, user1, becomes the device name. All other template and PKI certificate subject name information specific to the introduction is then provided by the user1 username records instead of by the administrator's record.

The registrar device uses the supplied username information with a user introducer name. The username allows the existing mechanisms for determining a user's authorization, template, and PKI certificate information to be supported without modification.

How Custom Templates Work with SDP

You may use custom templates to streamline the SDP process.

- Custom templates allow you to complete the web pages with the required start information, so the introducer is no longer required to contact the registrar and can immediately begin the SDP transaction.

- Custom templates allow customized deployment information to be displayed on the web pages, thereby tailoring the user experience.

An easy way to define a custom template is to modify the default template. Without custom templates, the introducer must contact the registrar for information to begin the SDP transaction. For a list of the default templates, see the section "Default Templates for SDP Transaction Web Pages, on page 138."

Note It is recommended that only advanced SDP users configure custom templates because problems can result from modifying templates incorrectly before the templates are displayed in the introducer's browser.

Custom Template Variable Expansion

There are expansion variables in the templates that are replaced by the Cisco IOS SDP registrar or petitioner. These variables are expanded as follows:

- $$--"$"
- $a--attribute-value (AV) pairs
- $c--Trusted certificate
- $d--Dump AV pairs in browser
- $h--Hostname
- $k--Keylabel or "tti"
- $l--Trustpoint label = "tti"
- $n--HTTP client's username
- $s--Default TTI key size
- $t--Trustpoint configuration
- $u--Completion URL
- $1 to $9--Variables retrieved from AAA server during user authentication

Custom Template Variable Expansion Rules

Configuration and templates are used during an SDP exchange. Prior to use and after distribution, these templates are expanded using the following rules based in the SDP communication stage.

Custom HTML Template Expansion Rules

HTML templates are expanded immediately before being served to the HTTP client. The HTTP templates are expanded as follows:

- $u--Completion url, which is be populated with the SDP completion URL (for example: http://10.10.10.1/ezsdd/completion). This variable is used internally by SDP as the internal "wizard" state. It is expected that the SDP introduction page include something similar to the following text: "<FORM action=\"$u\"method=\"post\">" for normal wizard processing.
- $n--introducer name or the device name entered by the administrative introducer.
- $$--$

- $h--Hostname
- $a--All AV pairs with or without a specified template character are written in the following HTML form format. (Because these AV pairs are not "INPUT type=hidden," they are directly displayed on the web page for debugging templates or the SDP process.)

<INPUT type=hidden NAME="attribute string here"

value="variable string here"><BR>

all HTML templates should have this!

$d = dump all av pairs in: attribute = value<BR>

URL Template Expansion Rules

There are URLs for the configuration template source, the file template source, and the file destination. These variables are expanded when the registrar prepares the URL, just before retrieving the configuration or file. For the file destination, these variables are expanded just before the petitioner copies the file to the file destination.

- $$--$
- $h--Hostname

URL Template Expansion Rules for iPhone Deployment

The following template expansion variables are introduced for iPhone deployment:

- $o - challenge password. This template character is expanded by the SDP registrar after it obtains the challenge password from the Simple Certificate Enrollment Protocol (SCEP) server, before the configuration profile is sent to the iPhone in the START phase.
- $i - unique device identifier (UDID) of the iPhone. This template character is expanded by the SDP registrar into the CN field of the Subject Name, before the configuration profile is sent to the iPhone in the INTRODUCTION phase.
- $p - subject name differentiator. This template character is expanded by the SDP registrar using the value configured through the CLI. Seethe Configuring the SDP Registrar to Deploy Apple iPhones, on page 155 for more information. This value can be used to differentiate the two certificates issued by the SCEP server to the iPhone, one in the COMPLETION phase and one in the VPN establishment phase. You determine part and field of the Subject Name into which this value goes.

See the How SDP Deploys Apple iPhones in a PKI, on page 141 for more information.

Custom Configuration and File Template Variable Expansion Rules

Custom configuration and file template variables are expanded both when the registrar prepares the configuration or file template and when the petitioner receives the configuration or file template.

Custom Configuration and File Template Variable Expansion Rules at the Registrar

When the registrar expands the configuration or file template, the following variables are used by the Cisco IOS CA. These variables are expanded before being sent through the SDP wizard.

- $$--$

- $h--Hostname
- $t--A simple default trustpoint configuration that includes $l, $k, and $s to be expanded at the client
- $1 to $9--Variables retrieved from AAA server during user authentication (not applicable to the file template)

Custom Configuration and File Template Variable Expansion Rules at the Petitioner

When the petitioner expands the configuration or file template, the following variables are expanded:

- $$--$
- $h--Hostname
- $k--Keylabel
- $l--Trustpoint label
- $s--Key size
- $c--Expanded to certificate chain
- $n--Expanded to username (not applicable to the file template)

Custom Configuration HTTP Template Variable Expansion Rules

Custom configuration HTTP templates provide flexibility for backend Common Gateway Interface (CGI) scripts and integration with external management systems. Template URLs run through the HTTP template expansions before registrar retrieves the bootstrap configuration from the external management system. The device name ($n) is expanded into the URL and passed to the external management system so that a specific bootstrap configuration file can be located based on the device information.

Note You should only modify the HTML text that is displayed. The existing expansion variables, Javascript, and forms in the default templates should not be removed when customizing the templates. They are required for SDP to function properly.

The HTTP template expansion and **template config** command allow you to specify either of the following file types to obtain a customized bootstrap configuration file:

- A configuration file based on the device name (for example, template config http://myserver/$n-config-file.conf*)*
- A CGI script based on the device name (for example, template config http://myserver/cgi-bin/mysdpcgi post)

As of Cisco IOS Release 12.4(6)T, the CGI support has been expanded so that the bootstrap configuration can be identified by not only the device name, but also the type, current Cisco IOS version information, and current configuration. This functionality expands the **template config** command with the **post** keyword, which tells the registrar to send this additional device information to the external management system through a CGI script with the HTTP or HTTPS protocol only.

The registrar passes the device information through AV pairs ($a) to the external management system. Using the AV pair information, the management system identifies the appropriate bootstrap configuration file and sends it back to the registrar. The additional AV pairs that are sent with the expanded CGI support for identification of the customized bootstrap configuration file are shown in the table below.

Table 5: AV Pairs Sent During HTTP Post to External Management System

AV Pair	Description
TTIFixSubjectName	AAA_AT_TTI_SUBJECTNAME (sent only if the realm authentication user is not the root user on the registrar)
TTIIosRunningConfig	Output of **show running-config brief**
TTIKeyHash	Digest calculated over the device public key
TTIPrivilege	AAA_AT_TTI_PRIVILEGE--”admin” is sent if the user is an administrator, “user” is sent if the user is not an administrator (sent only if the realm authentication user is an administrator and the information is available from the AAA server)
TTISignature	Digest calculated over all AV pairs except UserDeviceName and TTISignCert
TTISignCert	Device current certificate (sent only if the device currently has a certificate)
TTITemplateVar	AAA_AT_TTI_IOSCONFIG(1-9) (sent only if the realm authentication user is not the root user on the registrar)
TTIUserName	Device name
TTIVersion	TTI version of the registrar
UserDeviceName	Device name as entered by the administrative introducer (sent only if the realm authentication user is an administrator)

Note The registrar must be running Cisco IOS Release 12.4(6)T, the **template config** command must be issued with the **post** keyword, and the *url* argument must include either HTTP or HTTPS. No other protocol is supported for the expanded CGI template functionality (for example, FTP).

Default Templates for SDP Transaction Web Pages

The following default templates exist for each SDP transaction web page:

- Default Prep-Connect Template, on page 139
- Default Start Page Template, on page 140
- Default Welcome Page Template, on page 140
- Default Introduction Page Template, on page 141
- Default Admin-Introduction Page Template, on page 141
- Default Completion Page Template, on page 141

Default Prep-Connect Template

The prep-connect template may be modified by the administrator to contain values that are appropriate for their environment. The format of the prep-connect page may also be modified by the settings contained in the template.

Except for the registrar IP address, which the administrator must customize, the prep-connect template may be used as shown below.

```
<html><head><title>
SDP: Test Internet Connection</title></head>
<noscript><b>
If you see this message, your browser is not running JavaScript,<br>
which is required by Cisco Secure Device Provisioning.<br>
If you cannot enable JavaScript, please contact your system administrator.
<br><br></b></noscript>
<body style="background-color: rgb(204, 255, 255);">
<div style="text-align: center;"><big><big>
Secure Device Provisioning</big><br>
Test Internet Connection</big><br><br>
<form action="http://10.10.10.1/ezsdd/connect" method="post">
<input type="submit" value="Log onto Cisco Device"><br><br>
Default username/password is cisco/cisco.
<input type="hidden" name="TTIAfterConnectURL"
value="http://10.10.10.1/ezsdd/welcome">
<!-- Note, that for the below, 198.133.219.25 = www.cisco.com. -->
<input type="hidden" name="TTIConnectTestURL" value="http://198.133.219.25">
<input type="hidden" name="TTIInsideAddr" value="10.10.10.1">
<input type="hidden" name="TTIlanport" value="Vlan1">
<input type="hidden" name="TTIwanport" value="FastEthernet4">
</form></div></body></html>
```

Hidden HTML Form Fields

The hidden HTML form fields communicate initial configuration information to the browser as set by the administrator and are not signed.

Note The term “hidden” refers to the fact that these HTML form fields are not displayed on the prep-connect page to reduce potential confusion to the introducer.

The administrator can set hidden HTML form fields in the prep-connect template as shown in the table below.

Table 6: Administrator Defined AV Pairs Sent During Prep-Connect Phase

AV Pair	Description
TTIAfterConnectURL	The administrator may set the TTIAfterConnectURL field to either the welcome page URL or the start page URL. The welcome page URL is specified with the factory default petitioner IP address. The connect after URL may be any valid URL if SDP is not going to be used after establishing Internet connectivity.
TTIConnectTestURL	The administrator may set the TTIConnectTestURL field to a valid URL that should be accessible when Internet connectivity is established. The default prep-connect template value is www.cisco.com (198.133.219.25).

AV Pair	Description
TTIInsideAddr	The administrator may set the TTIInsideAddr field to the factory default IP address of the petitioner. For the Cisco 871 ISR, the IP address is 10.10.10.1.
TTIlanportx	The administrator may set the TTIlanportx field to the LAN interface name of the petitioner platform. This field is used to apply the Cisco IOS connect configuration. For the Cisco 871, the field value is “Vlan1.”
TTIwanport	The administrator may set the TTIwanport field to the WAN interface name of the petitioner. This field is used to apply the Cisco IOS connect configuration. For the Cisco 871, the field value is “FastEthernet4.”

Note The connect template cannot be customized.

Default Start Page Template

```
<html><head><title>EZ-Secure Device Deployment Start page on $h</title></head>
<NOSCRIPT><B>
If you see this message, your browser is not running JavaScript.<BR>
Cisco Secure Device Deployment requires JavaScript.<BR> Please contact
your system adminstrator.<BR><BR></B></NOSCRIPT>
<SCRIPT LANGUAGE="JavaScript">
function submit_to_url(form){
form.action=form.TTIWelcomeURL.value;return true;}</SCRIPT>
<B>Welcome to Cisco Secure Device Deployment Server $h</B> <FORM
action="" method="post" onSubmit="return submit_to_url(this)"> Your
device:<BR> <INPUT type="text" name="TTIWelcomeURL" size=80
value=\"\"><BR><BR> <INPUT type="submit" value="Next>"><BR>
$a</FORM></html>
```

Default Welcome Page Template

```
<html><head><title>EZ-Secure Device Deployment WELCOME to $h</title></head>
<NOSCRIPT><B>
If you see this message, your browser is not running JavaScript.<BR>
Cisco Secure Device Deployment requires JavaScript.<BR> Please contact
your system adminstrator.<BR><BR></B></NOSCRIPT>
<SCRIPT LANGUAGE=\"JavaScript\">
function submit_to_url(form){
natURL=location.href.split(\"/\");
localURL=form.TTICompletionURL.value.split(\"/\");
if(natURL[2]!=localURL[2]){
form.TTICompletionURL.value=localURL[0]+\"//\"+natURL[2]+\"/
\"+localURL[3]+
\"/\"+localURL[4];}
form.action=form.vpnserviceurl.value;
return true;}</SCRIPT>
<B>Welcome to Cisco Secure Device Deployment for $h</B> <FORM
action=\"\" method=\"post\" onSubmit=\"return submit_to_url (this)\">
To join a Virtual Private Network (VPN) enter the web<BR> site URL
provided by your network administrator:<BR> <INPUT type=\"text\"
name=\"vpnserviceurl\" size=80 value=\"\"><BR><BR><INPUT
type=\"submit\" value=\"Next>\"><BR> $a</FORM></html>
```

Default Introduction Page Template

```
<html><head><title>EZ-Secure Device Deployment INTRODUCTION to $h</title>
</head><B>Welcome to the VPN network gateway on $h</B> <FORM
action=\"$u\" method=\"post\"> Your 'username' and 'password' entered
have been accepted.<BR> Your device will now be allowed to
automatically join the VPN network.<BR> <BR>Press Next to complete
automatic configuration of your VPN Device.<BR> <BR><INPUT
type=\"submit\" value=\"Next>\"><BR> $a</P></FORM></html>
```

Default Admin-Introduction Page Template

```
<html><head><title>EZ-Secure Device Deployment ADMINISTRATIVE
INTRODUCTION to $h</title></head> <NOSCRIPT><B> If you see this
message, your browser is not running JavaScript.<BR> Cisco Secure
Device Deployment requires JavaScript.<BR> Please contact your system
adminstrator.<BR><BR></B></NOSCRIPT>
<SCRIPT LANGUAGE=\"JavaScript\">
function submit_to_url(form){
form.introadminurl.value=location.href+\"/admin\";
form.action=form.introadminurl.value;
return true;}</SCRIPT>
<B>Welcome to the VPN network gateway on $h</B> <FORM action=\"\"
method=\"post\" onSubmit=\"return submit_to_url (this)\"> Your
administrator 'username' and 'password' entered have been
accepted.<BR> Please provide the name to be associated with this
device:<BR> <INPUT type=\"text\" name=\"userdevicename\" size=64
value=\"\"><BR><BR> <INPUT type=\"submit\" value=\"Next>\"><BR> <INPUT
type=\"hidden\" name=\"introadminurl\" value=\"\"><BR>
$a</FORM></html>
```

Default Completion Page Template

```
<html><head><title>EZ-Secure Device Deployment COMPLETE on $h</title></head>
<B>Now enrolling $h with the VPN network...</B><BR> Full network VPN
access should be available in a moment.<BR><BR> $d<BR></html>
```

Default Template for the Configuration File

The default configuration template is shown below. This default configuration file is used if a configuration template is not specified or if the **template config** command is issued without the **post** keyword. For more information on using the default configuration template, see the UsingaConfigurationTemplateFile Example, on page 170.

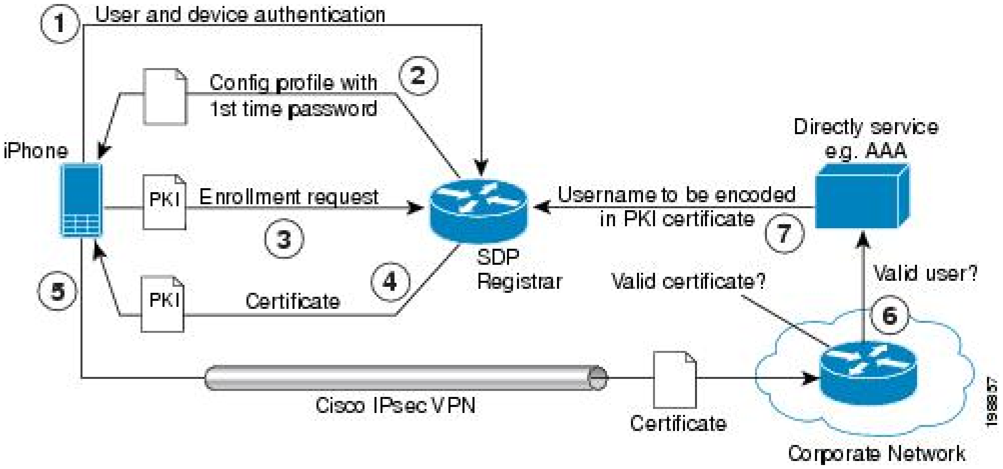
```
$t
!
$c
!
end
```

How SDP Deploys Apple iPhones in a PKI

With the introduction of the Cisco IOS 15.1(2)T and Apple iPhone OS 3.0 releases, Apple iPhones are supported on Cisco IOS network devices. Cisco IOS routers use the SDP registrar to deploy iPhones so that network applications can be accessed securely through an IPSec VPN, SCEP server, and PKI certificate deployment technologies.

The Apple iPhone combines the distribution of its XML-based “Configuration Profiles” with the initial deployment of certificates. SDP uses these initial certificates to authenticate access to enterprise applications and encrypt subsequent profile distribution. SDP uses this enrollment solution for distributing digital certificates to the iPhone.

Figure 17: SDP Registrar Deployment of the iPhone in a PKI



SDP Registrar Deployment Phases of the Apple iPhone in a PKI

The following sections describe each phase of the SDP registrar deployment of the iPhone in a PKI:

Start SDP Deployment Phase

The following steps describe the Start SDP deployment phase:

Note The Start SDP deployment phase is equivalent to the “Begin Enrollment” phase (or Phase 1) discussed in the http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf Apple iPhone Enterprise Deployment Guide .

SUMMARY STEPS

1. The iPhone user opens the Safari browser and types the start page HTTPS URL. For example, this HTTPS URL may be an internal corporate network address. The SDP registrar HTTPS page initiates the process.
2. The user starts authentication with the Cisco router, which acts as the SDP registrar by providing a username and password.
3. The SDP registrar contacts the SCEP server to obtain a challenge password.
4. The SDP registrar constructs a configuration profile in XML format that consists of the challenge password, SCEP server URL, and a request for iPhone attributes. The SCEP server URL is used to send the enrollment request and the iPhone device attributes are used by the iPhone to generate the RSA keys.
5. The iPhone user installs the configuration profile on the iPhone to complete the Start SDP phase.

DETAILED STEPS

Step 1 The iPhone user opens the Safari browser and types the start page HTTPS URL. For example, this HTTPS URL may be an internal corporate network address. The SDP registrar HTTPS page initiates the process.

Step 2 The user starts authentication with the Cisco router, which acts as the SDP registrar by providing a username and password.

Step 3 The SDP registrar contacts the SCEP server to obtain a challenge password.

Step 4 The SDP registrar constructs a configuration profile in XML format that consists of the challenge password, SCEP server URL, and a request for iPhone attributes. The SCEP server URL is used to send the enrollment request and the iPhone device attributes are used by the iPhone to generate the RSA keys.

The following example shows a configuration profile sent by the SDP registrar to the iPhone in the Start SDP deployment phase:

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Inc//DTD PLIST 1.0//EN" "http://
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>PayloadContent</key>
<dict>
<key>URL</key>
<string>https://profileserver.example.com/iphone</string>
<key>DeviceAttributes</key>
<array>
<string>UDID</string>
<string>IMEI</string>
<string>ICCID</string>
<string>VERSION</string>
<string>PRODUCT</string>
</array>
<key>Challenge</key>
<string>optional challenge</string>
```

Step 5 The iPhone user installs the configuration profile on the iPhone to complete the Start SDP phase.

Welcome SDP Deployment Phase

The Welcome SDP deployment phase is not applicable for the iPhone because the Introducer (for example, Safari web browser) is run on the SDP petitioner (iPhone).

Introduction SDP Deployment Phase

The following steps describe the Introduction SDP deployment phase:

Note The Introduction SDP deployment phase is equivalent to the “Device Authentication” phase (or Phase 2) discussed in the http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf Apple iPhone Enterprise Deployment Guide .

SUMMARY STEPS

1. The iPhone triggers an HTTPS post containing the requested device attribute information and the challenge password as a configuration profile. The HTTPS post is directed to the HTTPS URL specified in the configuration profile obtained in the Start SDP deployment phase, which must be the Introduction SDP deployment phase URL. The post data is signed by the iPhone using an Apple-issued certificate (built-in identity) and this signature may be verified, the identify confirmed, and the device attributes checked.
2. The UDID sent by the iPhone is captured by the SDP registrar and included in the Subject Name. Going forward, the device attributes obtained by the SDP registrar are used to determine if this was exactly the type of device that would be accepted. For example, the network administrator would only let 3GS iPhones onto the network because they have hardware encrypted storage. The device attributes obtained would enable the SDP registrar to distinguish 3GS iPhones from 3G iPhones.
3. The SDP registrar responds by building a configuration profile that consists of the following: HTTP URL of the SCEP server, Subject Name (contains the UDID) that is sent in the enrollment request, key size, key type, key usage, and challenge password. If the START phase had been skipped, the SDP registrar would contact the SCEP server to obtain a challenge password. See the URL Template Expansion Rules for iPhone Deployment, on page 136 for more information about how the SDP registrar obtains the Subject Name and the challenge password.

DETAILED STEPS

Step 1 The iPhone triggers an HTTPS post containing the requested device attribute information and the challenge password as a configuration profile. The HTTPS post is directed to the HTTPS URL specified in the configuration profile obtained in the Start SDP deployment phase, which must be the Introduction SDP deployment phase URL. The post data is signed by the iPhone using an Apple-issued certificate (built-in identity) and this signature may be verified, the identify confirmed, and the device attributes checked.

Step 2 The UDID sent by the iPhone is captured by the SDP registrar and included in the Subject Name. Going forward, the device attributes obtained by the SDP registrar are used to determine if this was exactly the type of device that would be accepted. For example, the network administrator would only let 3GS iPhones onto the network because they have hardware encrypted storage. The device attributes obtained would enable the SDP registrar to distinguish 3GS iPhones from 3G iPhones.

The following example shows a configuration profile sent by the iPhone in the Introduction SDP deployment phase:

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/
  DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>UDID</key>
<string></string>
<key>VERSION</key>
<string>7A182</string>
<key>MAC_ADDRESS_EN0</key>
<string>00:00:00:00:00:00</string>
<key>CHALLENGE</key>
either:
    <string>String</string>
or:
    <data>"base64 encoded data"</data>
</dict>
</plist>
```

Step 3 The SDP registrar responds by building a configuration profile that consists of the following: HTTP URL of the SCEP server, Subject Name (contains the UDID) that is sent in the enrollment request, key size, key type, key usage, and challenge password. If the START phase had been skipped, the SDP registrar would contact the SCEP server to obtain a challenge password. See the URL Template Expansion Rules for iPhone Deployment, on page 136 for more information about how the SDP registrar obtains the Subject Name and the challenge password.

Note The SDP registrar supports the RSA key type only.

The following example shows a configuration profile sent by the SDP registrar in the Introduction SDP deployment phase:

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Inc//DTD PLIST 1.0//EN" "http://
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>PayloadContent</key>
<dict>
<key>URL</key>
<string>https://iphone.vpn.apple.com/pkifoobar.exe</string>
<key>Name</key>
<string>instance_for_getcacert_call</string>
<key>Subject</key>
<array>
<array>
<array>
<string>O</string>
<string>Apple Inc.</string>
</array>
</array>
<array>
<array>
<string>CN</string>
<string>Foo</string>
</array>
</array>
</array>
<key>Challenge</key>
<string>CHALLENGE</string>
<key>Keysize</key>
<integer>1024</integer>
<key>Key Type</key>
<string>RSA</string>
<key>Key Usage</key>
<integer>5</integer>
</dict>
<key>PayloadDescription</key>
<string>Provides device encryption identity</string>
<key>PayloadUUID</key>
<string>fd8a6b9e-0fed-406f-9571-8ec98722b713</string>
<key>PayloadType</key>
<string>com.apple.security.scep</string>
<key>PayloadDisplayName</key>
<string>Encryption Identity</string>
<key>PayloadVersion</key>
<integer>1</integer>
<key>PayloadOrganization</key>
<string>Apple Inc.</string>
<key>PayloadIdentifier</key>
<string>com.apple.encrypted-profile-service</string>
```

```
</dict>
</plist>
```

Post-Introduction SDP Deployment Phase

The following steps describe the Post-introduction SDP deployment phase.

Note The Post-introduction SDP deployment phase is equivalent to the “Certificate Installation” phase (or Phase 3) discussed in the http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf Apple iPhone Enterprise Deployment guide .

SUMMARY STEPS

1. The iPhone installs the configuration profile specification containing SCEP information obtained from the SDP registrar in the Introduction SDP deployment phase.
2. The iPhone generates the keys with the instructions in the profile specification and sends the enrollment request to the SCEP server whose HTTP URL is specified in the profile, along with the challenge password.
3. The SCEP server verifies the challenge password and issues the digital certificate to the iPhone.
4. The user can install this certificate on the iPhone and use the Cisco IPsec VPN to connect to the corporate network.

DETAILED STEPS

Step 1 The iPhone installs the configuration profile specification containing SCEP information obtained from the SDP registrar in the Introduction SDP deployment phase.

Step 2 The iPhone generates the keys with the instructions in the profile specification and sends the enrollment request to the SCEP server whose HTTP URL is specified in the profile, along with the challenge password.

Step 3 The SCEP server verifies the challenge password and issues the digital certificate to the iPhone.

Step 4 The user can install this certificate on the iPhone and use the Cisco IPsec VPN to connect to the corporate network.

Note This certificate can also be used to download other enterprise settings, such as VPN settings, and Wi-Fi settings.

Second-Introduction SDP Deployment Phase

The following steps describe the Second-introduction SDP deployment phase:

Note The Second-introduction SDP deployment phase is equivalent to the “Device Configuration” phase (or Phase 4) discussed in the http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf Apple iPhone Enterprise Deployment guide .

SUMMARY STEPS

1. The iPhone repeats the Introduction SDP deployment phase with the following exceptions:
2. The SDP registrar responds with a configuration profile that includes the general enterprise settings such as VPN settings, Wi-Fi settings, and email settings. and in addition includes SCEP settings for a second certificate to be used for establishing a VPN.

DETAILED STEPS

Step 1 The iPhone repeats the Introduction SDP deployment phase with the following exceptions:

- The iPhone does not include the challenge password as part of the post data .
- The iPhone signs the post data using the certificate obtained from the SCEP server in the Post-introduction SDP deployment phase.

Step 2 The SDP registrar responds with a configuration profile that includes the general enterprise settings such as VPN settings, Wi-Fi settings, and email settings. and in addition includes SCEP settings for a second certificate to be used for establishing a VPN.

Second Post-Introduction SDP Deployment Phase

The Second Post-introduction SDP phase is identical to the Post-introduction SDP deployment phase. The iPhone generates a certificate request based on the SCEP settings provided by the SDP registrar in the Second-introduction SDP deployment phase and enrolls with the SCEP server.

Completion SDP Deployment Phase

The Completion SDP deployment phase is not applicable for the iPhone because the Introducer (for example, the Safari web browser) is run on the SDP petitioner (iPhone).

How to Set Up Secure Device Provisioning (SDP) for Enrollment in a PKI

This section contains the following procedures that should be followed when setting up SDP for your PKI. You can configure the registrar according to only one of the registrar configuration tasks.

Enabling the SDP Petitioner

Perform this task to enable or disable the petitioner and associate a trustpoint with the SDP exchange.

You can also use this task to configure the petitioner to use a certificate and the RSA keys associated with a specific trustpoint.

Note The petitioner is enabled by default on a Cisco device that contains a crypto image; thus, you have only to issue the **crypto provisioning petitioner** command if you have previously disabled the petitioner or if you want to use an existing trustpoint instead of the automatically generated trustpoint.

Note By default, the SDP petitioner device uses an existing certificate. If multiple certificates and one specific certificate exist, use this task to make a choice. However, this task is not necessary to enable the default behavior.

Before you begin

- The HTTP server must be enabled through the **ip http server**command. (The HTTP server is typically enabled by default in many default Cisco IOS configurations.)
- If you are configuring the petitioner to use a certificate and RSA keys, your SDP petitioner device must have an existing manufacturer's certificate or a third-party certificate.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto provisioning petitioner**
4. Do one of the following:
   - **trustpoint** *trustpoint-label*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**crypto provisioning petitioner**
	Example:
	`Router(config)#`
	` crypto provisioning petitioner`
		Allows SDP petitioner device behavior to be modified and enters tti-petitioner configuration mode.
		Note Effective with Cisco IOS Release 12.3(14)T, the **crypto provisioning petitioner**command replaced the **crypto wui tti petitioner**command.

	Command or Action	Purpose

Step 4	Do one of the following:
	• **trustpoint** *trustpoint-label*
	Example:
	`Router(tti-petitioner)# trustpoint mytrust`
	Example:
	Example:
	Example:
	`trustpoint signing`
	`trustpoint-label`
	Example:
	`Router(tti-petitioner)# trustpoint signing mytrust`
		(Optional) Specifies the trustpoint that is to be associated with the SDP exchange between the petitioner and the registrar.
		Note If this command is not issued, the *trustpoint-label*argument is aut omatically labeled “tti.”
		(Optional) Specifies the trustpoint and associated certificate that are used when signing all introduction data during the SDP exchange.

Step 5	**end**
	Example:
	`Router(tti-petitioner)# end`
		(Optional) Exits tti-petitioner configuration mode.

Troubleshooting Tips

After the SDP exchange is complete, a new trustpoint-label named “tti” exists. The trustpoint is automatically enrolled with the certificate server (the registrar). To verify that the trustpoint is really there, use the **show running-config** command.

What to Do Next

If you set up the petitioner to use a certificate and the RSA keys associated with the specified trustpoint, you should configure the registrar as shown in the task “Enabling the SDP Registrar for Certificate-Based Authorization.”

Enabling the SDP Registrar and Adding AAA Lists to the Server

Perform this task to enable the registrar and associate a certificate server with the SDP exchange.

You can also use this task if you want to add an authentication list and an authorization list to the RADIUS or TACACS+ server.

Prerequisites

Before configuring a registrar, perform the following tasks:

- Enable the HTTP server or the HTTPS server.

Note Before you enable an HTTPS server, you must disable the standard HTTP server if it is configured. Use the the **no ip http server** command to disable an HTTP server. To enable an HTTPS server, you should issue the **ip http secure-server** command followed by the **ip http secure-trustpoint** command. The specified trustpoint is a registrar local trustpoint appropriate for HTTPS communication between the registrar and the user's browser.

- Configure the Cisco IOS certificate server through the **crypto pki server** command.

If you are configuring AAA lists, you should complete the prerequisites required for the registrar in addition to completing the following tasks:

- Add user information to the AAA server database. To configure a RADIUS or TACACS+ AAA server, see the "Configuring RADIUS" and "Configuring TACACS+ " chapters of t he *Cisco IOS Security Configuration Guide* .
- Configure new AAA lists. To configure AAA lists, see the following chapters in the *Cisco IOS Security Configuration Guide* : "Configuring RADIUS," "Configuring TACACS+," "Configuring Authentication," and "Configuring Authorization ."

Restrictions

Cisco IOS CA Device Requirement

During the SDP process, a Cisco IOS CA certificate is automatically issued to the peer device. If an SDP registrar is configured on a third-party vendor's CA device, the SDP process does not work.

The template config Command

There are nine Cisco IOS configuration variables. If you require more configuration flexibility, the **template config** command can be used to reference a configuration template that is specific to the introducer. For more information on configuration flexibility, see the "Custom Configuration and File Template Variable Expansion Rules, on page 136" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto provisioning registrar**
4. **pki-server** *label*
5. **authentication list** *list-name*
6. **authorization list** *list-name*
7. **template username** *name* **password** *password*
8. **template config** *url* [**post**]

9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto provisioning registrar** **Example:** `Router(config)# crypto provisioning registrar`	Configures a device to become a registrar for the SDP exchange and enters tti-registrar configuration mode. **Note** Effective with Cisco IOS Release 12.3(14)T, the **crypto provisioning registrar**command replaced the **crypto wui tti registrar**command.
Step 4	**pki-server** *label* **Example:** `Router(tti-registrar)# pki-server mycs`	Specifies the certificate server that is to be associated with the SDP exchange between the petitioner and the registrar.
Step 5	**authentication list** *list-name* **Example:** `Router (tti-registrar)# authentication list authen-tac`	(Optional) Authenticates the introducer in an SDP exchange.
Step 6	**authorization list** *list-name* **Example:** `Router (tti-registrar)# authorization list author-rad`	(Optional) Receives the appropriate authorized fields for the certificate subject name and list of template variables to be expanded into the Cisco IOS CLI snippet that is sent back to the petitioner.
Step 7	**template username** *name* **password** *password* **Example:** `Router(tti-registrar)# template username ftpuser password ftppwd`	(Optional) Establishes a username and password in which to access the configuration template on the file system.
Step 8	**template config** *url* [**post**] **Example:** `Router(tti-registrar)# template config http://myserver/cgi-bin/mycgi post`	(Optional) Specifies a remote URL for the Cisco IOS CLI configuration template. The *url* argument can reference a configuration file that allows you to specify the device name ($n) to identify a bootstrap configuration. CGI support allows you to

	Command or Action	Purpose
		reference a CGI script through either HTTP or HTTPS and identify the bootstrap configuration by not only the device name, but also by the type, current Cisco IOS version and current configuration. The **post** keyword must be used for CGI support. **Note** The registrar must be running Cisco IOS Release 12.4(6)T or later to utilize expanded CGI support. If the registrar is running an earlier version of Cisco IOS, the additional device identification information is ignored.

Step 9	**end**
	Example:
	`Router(tti-registrar)# end`
		(Optional) Exits tti-registrar configuration mode.

Examples

To help troubleshoot the SDP transaction, you can issue the **debug crypto provisioning**command, which displays output from the petitioner and registrar devices.

The following is output for the **debug crypto provisioning**command. The output from the petitioner and registrar devices are shown below.

```
Petitioner device
! The user starts the Welcome phase.
Nov  7 03:15:48.171: CRYPTO_PROVISIONING: received welcome get request.
! The router generates a Rivest, Shamir, and Adelman (RSA) keypair for future enrollment.
Nov  7 03:15:48.279: CRYPTO_PROVISIONING: keyhash 'A506BE3B83C6F4B4A6EFCEB3D584AACA'
! The TTI transaction is completed.
Nov  7 03:16:10.607: CRYPTO_PROVISIONING: received completion post request.
Registrar device
!. During the introduction phase, the browser prompts for login information.
06:39:18: CRYPTO_PROVISIONING: received introduction post request.
06:39:18: CRYPTO_PROVISIONING: checking AAA authentication (ipsecca_script_aaalist, ttiuser)
! This happens if the user types in the wrong username or password.
06:39:19: CRYPTO_PROVISIONING: authentication declined by AAA, or AAA server not found -
0x3
06:39:19: CRYPTO_PROVISIONING: aaa query fails!
! The user re-enters login information.
06:39:19: CRYPTO_PROVISIONING: received introduction post request.
06:39:19: CRYPTO_PROVISIONING: checking AAA authentication (ipsecca_script_aaalist, ttiuser)
06:39:20: CRYPTO_PROVISIONING: checking AAA authorization (ipsecca_script_aaalist, ttiuser)
! The login attempt succeeds and authorization information is retrieved from the AAA database.
06:39:21: CRYPTO_PROVISIONING: aaa query ok!
! These attributes are inserted into the configuration template.
06:39:21: CRYPTO_PROVISIONING: building TTI av pairs from AAA attributes
06:39:21: CRYPTO_PROVISIONING: "subjectname" = "CN=user1, O=company, C=US"
06:39:21: CRYPTO_PROVISIONING: "$1" = "ntp server 10.3.0.1"
06:39:21: CRYPTO_PROVISIONING: "$2" = "hostname user1-vpn"
! The registrar stores this subject name and overrides the subject name in the subsequent
enrollment request.
06:39:21: CRYPTO_PROVISIONING: subjectname=CN=user1, O=company, C=US
```

```
! The registrar stores this key information so that it may be used to automatically grant
the subsequent enrollment request.
06:39:21: CRYPTO_PROVISIONING: key_hash=A506BE3B83C6F4B4A6EFCEB3D584AACA
```

Enabling the SDP Registrar for Certificate-Based Authorization

Perform this task to enable the SDP registrar to verify the petitioner-signing certificate using either a specified trustpoint or any configured trustpoint and initiate authorization lookups using the introducer username and the certificate name field.

Before you begin

You must also configure the SDP petitioner to use a certificate and RSA keys associated with a specific trustpoint. To complete this task, use the trustpoint signing command as shown in the task "Enabling the SDP Petitioner, on page 147."

Note Because RADIUS does not differentiate between authentication and authorization, you need to use the default password, cisco, for certificate authorization.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto provisioning registrar**
4. **template file** *sourceURL destinationURL*
5. **binary file** *sourceURL destinationURL*
6. **authentication trustpoint** {**trustpoint-label| use-any** }
7. **authorization {login | certificate | login certificate}**
8. **authorization username subjectname** *subjectname*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**crypto provisioning registrar** **Example:** `Router(config)# crypto provisioning registrar`	Configures a device to become an SDP registrar and enters tti-registrar configuration mode.
Step 4	**template file** *sourceURL destinationURL* **Example:** `Router(tti-registrar)# template file http://myserver/registrar_file_r1 http://myserver/petitioner_file_p1`	(Optional) Specifies the source template file location on the registrar and the destination template file location on the petitioner. **Note** This command is useful when using a USB token to provision a device. The template expansion occurs on the registrar for both the source URL and file content. The destination URL is expanded on the petitioner.
Step 5	**binary file** *sourceURL destinationURL* **Example:** `Router(tti-registrar)# binary file http://myserver/registrar_file_a1 http://myserver/petitioner_file_b1`	(Optional) Specifies the binary file location on the registrar and the destination binary file location on the petitioner. **Note** This command is useful when using a USB token to provision a device. Both the source and destination URL are expanded on the registrar. Also, the destination URL and file content are expanded on the petitioner. Binary files are not processed through the template expansion functions.

Step 6	**authentication trustpoint** {**trustpoint-label**| **use-any** }
	Example:
	`Router(tti-registrar)# authentication trustpoint mytrust`
		(Optional) Specifies the trustpoint used to authenticate the SDP petitioner device’s existing certificate.
		• *trustpoint-label* --Specifies a specific trustpoint.
		• **use-any** --Specifies any configured trustpoint.
		Note If you do not use this command to specify a trustpoint, the existing petitioner certificate is not validated. (This functionality provides compatibility with self-signed petitioner certificates.)

Step 7	**authorization {login | certificate | login certificate}**
	Example:
	`Router(tti-registrar)# authorization login certificate`
		(Optional) Enables AAA authorization for an introducer or a certificate.
		• Use the **login** keyword for authorization based on the introducer’s username.
		• Use the **certificate** keyword for authorization based on the petitioner’s certificate.
		• Use the **login certificate** keyword for authorization based on the introducer’s username and the petitioner’s certificate.

	Command or Action	Purpose
Step 8	**authorization username subjectname** *subjectname* **Example:** `Router(tti-registrar)# authorization username subjectname all`	Sets parameters for the different certificate fields that are used to build the AAA username. • The **all** keyword specifies that the entire subject name if the certificate is used as the authorization username.

Step 9	**end**
	Example:
	`Router(tti-registrar)# end`
		(Optional) Exits tti-registrar configuration mode.

Configuring the SDP Registrar to Deploy Apple iPhones

Perform this task to configure the SDP registrar to run HTTPS in order to deploy Apple iPhones on a corporate network.

Before you begin

Ensure that the SDP Registrar is enabled to run HTTPS. See the Enabling the SDP Registrar and Adding AAA Lists to the Server section for more information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http secure-server**
4. **crypto provisioning registrar**
5. **url-profile start** *profile-name*
6. **url-profile intro** *profile-name*
7. **match url** *url*
8. **match authentication trustpoint** *trustpoint-name*
9. **match certificate** *certificate-map*
10. **mime-type** *mime-type*
11. **template location** *location*
12. **template variable p** *value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip http secure-server** **Example:** `Router(config)# ip http secure-server`	Enables the HTTPS web server.
Step 4	**crypto provisioning registrar** **Example:** `Router(config)# crypto provisioning registrar`	Configures a device to become a registrar for the SDP exchange and enters tti-registrar configuration mode. **Note** Effective with Cisco IOS Release 12.3(14)T, the **crypto provisioning registrar**command replaced the **crypto wui tti registrar**command.
Step 5	**url-profile start** *profile-name* **Example:** `Router(tti-registrar)# url-profile start START`	Specifies the **start** keyword to indicate that a URL profile is to be associated with the Start SDP deployment phase. The *profile-name* argument specifies the name of a unique URL profile. **Note** Both the Introduction SDP deployment phase and the Start SDP deployment phase can use different profiles or use the same URL profile.
Step 6	**url-profile intro** *profile-name* **Example:** `Router(tti-registrar)# url-profile intro INTRO`	Specifies the **intro** keyword to indicate that a URL profile is to be associated with the Introduction SDP deployment phase. The *profile-name* argument specifies the name of a unique URL profile. **Note** Both the Introduction SDP deployment phase and the Start SDP deployment phase can use different profiles or use the same URL profile.
Step 7	**match url** *url* **Example:** `Router(tti-registrar)# match url /sdp/intro`	Specifies the URL to be associated with the URL profile.

Step 8	**match authentication trustpoint** *trustpoint-name*
	Example:
	`Router(tti-registrar)# match authentication`
	`trustpoint apple-tp`
		(Optional) Specifies the trustpoint name that should be used to authenticate the peer's certificate. If the trustpoint name is not specified, then the trustpoint configured using the **authentication trustpoint command**in tti-registrar configuration mode is used to authenticate the peer's certificate. See the Enabling the SDP Registrar for Certificate-Based Authorization section for more information.

	Command or Action	Purpose
Step 9	**match certificate** *certificate-map* **Example:** `Router(tti-registrar)# match certificate cat 10`	(Optional) Specifies the name of the certificate map used to authorize the peer's certificate.
Step 10	**mime-type** *mime-type* **Example:** `Router(tti-registrar)# mime-type application/x-apple-aspen-config`	Specifies the Multipurpose Internet Mail Extensions (MIME) type that the SDP registrar should use to respond to a request received through this URL profile.
Step 11	**template location** *location* **Example:** `Router(tti-registrar)# template location flash:intro.mobileconfig`	Specifies the location of the template that the SDP Registrar should use while responding to a request received through this URL profile.
Step 12	**template variable p** *value* **Example:** `Router(tti-registrar)# template variable p iphone-vpn`	(Optional) Specifies the value that goes into the Organizational Unit (OU) field of the subject name in the trustpoint certificate to be issued by the SDP Registrar. See this field in the certificate presented in the Apple CA Server Trustpoint Certificate Configuration Example section below.

Apple CA Server Trustpoint Certificate Configuration

The SDP Registrar must verify the signature generated from the iPhone's trustpoint certificate in order to trust the Apple CA server certificate. The iPhone signs its messages using the trustpoint certificate, which is issued by Apple's CA server during the Introduction SDP deployment phase.

The following example shows how to configure certificate enrollment using the manual cut-and-paste enrollment method of the Apple CA certificate:

Note See also the "How to Configure Certificate Enrollment for a PKI" section in the Configuring Certificate Enrollment for a PKI feature module for more detailed information about configuring a trustpoint certificate.

SUMMARY STEPS

1. The**crypto pki trustpoint** command is entered in global configuration mode to declare the trustpoint and a given name and enters ca-trustpoint configuration mode:
2. The **enrollment terminal** command is entered to specify manual cut-and-paste certificate enrollment
3. The **crypto pki authenticate** command retrieves the CA certificate and authenticates it from the specified TFTP server.
4. Copy the following block of text containing the base 64 encoded Apple CA trust certificate and paste it at the prompt.
5. The **exit** command is used to exit ca-trustpoint configuration mode and enter global configuration mode.

6. The **crypto provisioning registrar** command is entered in global configuration mode to specify the router to become a registrar for the SDP exchange and enters tti-registrar configuration mode.
7. The **url-profile command with the intro** keyword is entered in tti-registrar configuration mode to specify the unique URL profile name that is associated with the Introduction SDP deployment phase.
8. The **match authentication trustpoint**command is entered in tti-registrar configuration mode to specify the trustpoint name that should be used to authenticate the peer’s certificate.

DETAILED STEPS

Step 1 The**crypto pki trustpoint** command is entered in global configuration mode to declare the trustpoint and a given name and enters ca-trustpoint configuration mode:

Example:

```
Router(config)# crypto pki trustpoint apple-tp
```

Step 2 The **enrollment terminal** command is entered to specify manual cut-and-paste certificate enrollment

Example:

```
Router(ca-trustpoint)# enrollment terminal
```

Step 3 The **crypto pki authenticate** command retrieves the CA certificate and authenticates it from the specified TFTP server.

Example:

```
Router(ca-trustpoint)# crypto pki authenticate apple-tp
```

Step 4 Copy the following block of text containing the base 64 encoded Apple CA trust certificate and paste it at the prompt.

Example:

```
I Bag Attributes
    localKeyID: 7C 29 15 15 12 C9 CF F6 15 2B 5B 25 70 3D A7 9A 98 14 36 06
subject=/C=US/O=Apple Inc./OU=Apple iPhone/CN=Apple iPhone Device CA
issuer=/C=US/O=Apple Inc./OU=Apple Certification Authority/CN=Apple iPhone Certification Authority
-----BEGIN CERTIFICATE-----
MIIDaTCCAlGgAwIBAgIBATANBgkqhkiG9w0BAQUFADB5MQswCQYDVQQGEwJVUzET
MBEGA1UEChMKQXBwbGUgSW5jLjEmMCQGA1UECxMdQXBwbGUgQ2VydGlmaWNhdGlv
biBBdXRob3JpdHkxLTArBgNVBAMTJEFwcGxlIGlQaG9uZSBDZXJ0aWZpY2F0aW9u
IEF1dGhvcml0eTAeFw0wNzA0MTYyMjU0NDZaFw0xNDA0MTYyMjU0NDZaMFoxCzAJ
BgNVBAYTAlVTMRMwEQYDVQQKEwpBcHBsZSBJbmMuMRUwEwYDVQQLEwxBcHBsZSBp
UGhvbmUxHzAdBgNVBAMTFkFwcGxlIGlQaG9uZSBEZXZpY2UgQ0EwgZ8wDQYJKoZI
hvcNAQEBBQADgY0AMIGJAoGBAPGUSsnquloYYK3Lok1NTlQZaRdZB2bLl+hmmkdf
Rq5nerVKc1SxywT2vTa4DFU4ioSDMVJl+TPhl3ecK0wmsCU/6TKqewh0lOzBSzgd
Z04IUpRai1mjXNeT9KD+VYW7TEaXXm6yd0UvZ1y8Cxi/WblshvcqdXbSGXH0KWO5
JQuvAgMBAAGjgZ4wgZswDgYDVR0PAQH/BAQDAgGGMA8GA1UdEwEB/wQFMAMBAf8w
HQYDVR0OBBYEFLL+ISNEhpVqedWBJo5zENinTI50MB8GA1UdIwQYMBaAFOc0Ki4i
3jlga7SUzneDYS8xoHw1MDgGA1UdHwQxMC8wLaAroCmGJ2h0dHA6Ly93d3cuYXBw
bGUuY29tL2FwcGxlY2EvaXBob25lLmNybDANBgkqhkiG9w0BAQUFAAOCAQEAd13P
Z3pMViukVHe9WUg8Hum+0I/0kHKvjhwVd/IMwGlXyU7DhUYWdja2X/zqj7W24Aq5
7dEKm3fqqxK5XCFVGY5HI0cRsdENyTP7lxSiiTRYj2mlPedheCn+k6T5y0U4Xr40
FXwWb2nWqCF1AgIudhgvVbxlvqcxUm8Zz7yDeJ0JFovXQhyO5fLUHRLCQFssAbf8
B4i8rYYsBUhYTspVJcxVpIIltkYpdIRSIARA49HNvKK4hzjzMS/OhKQpVKw+OCEZ
xptCVeN2pjbdt9uzi175oVo/u6B2ArKAW17u6XEHIdDMOe7cb33peVI6TD15W4MI
pyQPbp8orlXe+tA8JA==
-----END CERTIFICATE-----
```

Step 5 The **exit** command is used to exit ca-trustpoint configuration mode and enter global configuration mode.

Example:

```
Router(ca-trustpoint)# exit
```

Step 6 The **crypto provisioning registrar** command is entered in global configuration mode to specify the router to become a registrar for the SDP exchange and enters tti-registrar configuration mode.

Example:

```
Router(config)# crypto provisioning registrar
```

Step 7 The **url-profile command with the intro** keyword is entered in tti-registrar configuration mode to specify the unique URL profile name that is associated with the Introduction SDP deployment phase.

Example:

```
Router(tti-registrar)# url-profile intro INTRO
```

Step 8 The **match authentication trustpoint**command is entered in tti-registrar configuration mode to specify the trustpoint name that should be used to authenticate the peer's certificate.

Example:

```
Router(tti-registrar)# match authentication trustpoint apple-tp
```

The SDP Registrar can now use the Apple CA trustpoint certificate called "apple-tp" for verifying the signature of the iphone.

Configuring an Administrative Introducer

Perform the following task to configure an administrative introducer using administrator authentication and authorization lists.

Before you begin

The administrative introducer must have enable privileges on the client device and administrator privileges on the server.

Note When using RADIUS, a user/device that needs to be introduced by the administrative introducer must always use cisco as its own password. TACACS+ does not have this limitation; a user/device can have any password and be introduced by the administrative introducer.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **crypto provisioning registrar**
4. **administrator authentication list** *list-name*
5. **administrator authorization list** *list-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto provisioning registrar** **Example:** `Router(config)# crypto provisioning registrar`	Configures a device to become an SDP registrar and enters tti-registrar configuration mode.
Step 4	**administrator authentication list** *list-name* **Example:** `Router(tti-registrar)# administrator authentication list authen-tac`	Configures the AAA list used to authenticate an administrator during an introduction.
Step 5	**administrator authorization list** *list-name* **Example:** `Router(tti-registrar)# administrator authorization list author-tac`	Configures the AAA list used to obtain authorization information for an administrator during an introduction. Information that can be obtained includes the certificate subject name and/or the list of template variables to be expanded into the Cisco IOS CLI snippet that is sent back to the petitioner.

Step 6	**end**
	Example:
	`Router(tti-registrar)# end`
		(Optional) Exits tti-registrar configuration mode.

Example

The following example from the **show running-config**command allows you to verify that an administrative introducer using administrator authentication and authorization lists have been created:

```
Router# show running-config
Building configuration...
```

```
Current configuration : 2700 bytes
!
! Last configuration change at 01:22:26 GMT Fri Feb 4 2005
!
version 12.4
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname router
!
boot-start-marker
boot-end-marker
!
memory-size iomem 5
enable secret 5 $1$tpBS$PXnBDTIDXfX5pWa//1JX20
enable password lab
!
aaa new-model
!
!
!
aaa session-id common
!
resource manager
!
clock timezone GMT 0
ip subnet-zero
no ip routing
!
!
no ip dhcp use vrf connected
!
!
no ip cef
no ip domain lookup
ip domain name company.com
ip host router 10.3.0.6
ip host router.company.com 10.3.0.6
no ip ips deny-action ips-interface
!
no ftp-server write-enable
!
crypto pki server mycs
!
crypto pki trustpoint mycs
 revocation-check crl
 rsakeypair mycs
!
crypto pki trustpoint tti
 revocation-check crl
 rsakeypair tti
!
crypto pki trustpoint mic
 enrollment url http://router:80
 revocation-check crl
!
crypto pki trustpoint cat
 revocation-check crl
!
!
!
crypto pki certificate map cat 10
```

```
!
crypto pki certificate chain mycs
 certificate ca 01
crypto pki certificate chain tti
crypto pki certificate chain mic
 certificate 02
 certificate ca 01
crypto pki certificate chain cat
!
crypto provisioning registrar <---------- !SDP registrar device parameters!
 administrator authentication list authen-tac
 administrator authorization list author-tac
!
no crypto engine onboard 0
username qa privilege 15 password 0 lab
```

Configuring Custom Templates

Perform this task to create and configure custom templates.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto provisioning registrar**
4. **template http start** *URL*
5. **template http welcome** *URL*
6. **template http introduction** *URL*
7. **template http admin-introduction** *URL*
8. **template http completion** *URL*
9. **template http error** *URL*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto provisioning registrar** **Example:** `Router(config)# crypto provisioning registrar`	Configures a device to become an SDP registrar and enters tti-registrar configuration mode.

	Command or Action	Purpose

Step 4	**template http start** *URL*
	Example:
	`Router(tti-registrar)# template http start tftp://`
	`registrar.company`
	`.com/start.html`
		Directs the TTI registrar to use the custom start page template.
		Note This command is required to use the start page functionality. If this command is not issued, the welcome page is the initial communication between the introducer and the petitioner.

Step 5	**template http welcome** *URL*
	Example:
	`Router(tti-registrar)#`
	`template http welcome`
	`tftp://registrar.company.com/welcome.html`
		(Optional) Uses a custom welcome template rather than the default template.

Step 6	**template http introduction** *URL*
	Example:
	`Router(tti-registrar)#`
	`template http introduction`
	`tftp://registrar.company.com/intro.html`
		(Optional) Uses a custom introduction template rather than the default template.

Step 7	**template http admin-introduction** *URL*
	Example:
	`Router(tti-registrar)#`
	`template http admin-introduction`
	`tftp://registrar.company.com/admin-intro.html`
		(Optional) Uses a custom admin-introduction template rather than the default template.

Step 8	**template http completion** *URL*
	Example:
	`Router(tti-registrar)#`
	`template http completion`
	`tftp://registrar.company.com/completion.html`
		(Optional) Uses a custom completion template rather than the default template.

Step 9	**template http error** *URL*
	Example:
	`Router(tti-registrar)#`
	`template http error`
	`tftp://registrar.company.com/error.html`
		(Optional) Uses a custom error template rather than the default template.

Step 10	**end**
	Example:
	`Router(tti-registrar)# end`
		(Optional) Exits tti-registrar configuration mode.

Example

The following example shows the use of custom start, introduction, and completion templates:

template http start tftp://registrar.company.com/start.html

template http introduction tftp://registrar.company.com/intro.html

template http completion tftp://registrar.company.com/completion.html

Configuration Examples for Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI

Verifying the SDP Registrar Example

The following sample output from the **show running-config**command verifies that the certificate server “cs1” was configured and associated with the SDP exchange between the registrar and petitioner:

```
Router# show running-config
Building configuration...
Current configuration : 5902 bytes
!
! Last configuration change at 09:34:44 GMT Sat Jan 31 2004
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname pki-36a
!
boot-start-marker
boot-end-marker
!
logging buffered 32768 debugging
no logging console
enable secret 5 $1$b3jz$CKquLGjFIE3AdXA2/Rl9./
enable password lab
!
clock timezone GMT 0
no aaa new-model
ip subnet-zero
!
!
ip cef
ip domain name company.com
ip host msca-root
ip host yni-u10
ip host pki-36a 10.23.2.131
ip host pki-36a.company.com 10.23.2.131
!
!
crypto pki server cs1
 issuer-name CN=company,L=city,C=US
 hash sha1
 lifetime crl 336
 lifetime certificate 730
!
crypto pki trustpoint pki-36a
 enrollment url http://pki-36a:80
 ip-address FastEthernet0/0
 revocation-check none
!
crypto pki trustpoint cs1
 revocation-check crl
 rsakeypair cs1 2048
!
!
crypto pki certificate chain pki-36a
 certificate 03
  308201D0 30820139 A0030201 02020103 300D0609 2A864886 F70D0101 04050030
  34310B30 09060355 04061302 55533114 30120603 55040713 0B205361 6E746120
  4372757A 310F300D 06035504 03130620 696F7363 73301E17 0D303430 31333130
  39333334 345A170D 30363031 33303039 33333434 5A303A31 38301606 092A8648
  86F70D01 09081309 31302E32 332E322E 32301E06 092A8648 86F70D01 09021611
  706B692D 3336612E 63697363 6F2E636F 6D305C30 0D06092A 864886F7 0D010101
  0500034B 00304802 4100AFFA 8F429618 112FAB9D 01F3352E 59DD3D2D AE67E31D
  370AC4DA 619735DF 9CF4EA13 64E4B563 C239C5F0 1578B773 07BED641 A18CA629
  191884B5 61B66ECF 4D110203 010001A3 30302E30 0B060355 1D0F0404 030205A0
  301F0603 551D2304 18301680 141DA8B1 71652961 3F7D69F0 02903AC3 2BADB137
  C6300D06 092A8648 86F70D01 01040500 03818100 67BAE186 327CED31 D642CB39
  AD585731 95868683 B950DF14 3BCB155A 2B63CFAD B34B579C 79128AD9 296922E9
  4DEDFCAF A7B5A412 AB1FC081 09951CE3 08BFFDD9 9FB1B9DA E9AA42C8 D1049268
  C524E58F 11C6BA7F C750320C 03DFB6D4 CBB3E739 C8C76359 CE939A97 B51B3F7F
  3FF;A9D82 9CFDB6CF E2503A14 36D0A236 A1CCFEAE
  quit
 certificate ca 01
  30820241 308201AA A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  34310B30 09060355 04061302 55533114 30120603 55040713 0B205361 6E746120
  4372757A 310F300D 06035504 03130620 696F7363 73301E17 0D303430 31333130
  39333132 315A170D 30373031 33303039 33313231 5A303431 0B300906 03550406
  13025553 31143012 06035504 07130B20 53616E74 61204372 757A310F 300D0603
  55040313 0620696F 73637330 819F300D 06092A86 4886F70D 01010105 0003818D
  00308189 02818100 FC0695AF 181CE90A 1B34B348 BA957178 680C8B51 07802AC3
  BF77B9C6 CB45092E 3C22292D C7D5FFC1 899185A1 FD8F37D5 C44FC206 6D1FA581
  E2264C83 1CC7453E 548C89C6 F3CD25BC 9BFFE7C5 E6653A06 62133950 78BED51B
  49128428 AB237F80 83A530EA 6F896193 F2134B54 D181F059 348AA84B 21EE6D80
  727BF668 EB004341 02030100 01A36330 61300F06 03551D13 0101FF04 05300301
  01FF300E 0603551D 0F0101FF 04040302 0186301D 0603551D 0E041604 141DA8B1
  71652961 3F7D69F0 02903AC3 2BADB137 C6301F06 03551D23 04183016 80141DA8
  B1716529 613F7D69 F002903A C32BADB1 37C6300D 06092A86 4886F70D 01010405
  00038181 00885895 A0141169 3D754EB2 E6FEC293 5BF0A80B E424AA2F A3F59765
  3463AAD1 55E71F0F B5D1A35B 9EA79DAC DDB40721 1344C01E 015BAB73 1E148E03
  9DD01431 A5E2887B 4AEC8EF4 48ACDB66 A6F9401E 8F7CA588 8A4199BB F8A437A0
  F25064E7 112805D3 074A154F 650D09B9 8FA19347 ED359EAD 4181D9ED 0C667C10
  8A7BCFB0 FB
  quit
crypto pki certificate chain cs1
 certificate ca 01
  30820241 308201AA A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  34310B30 09060355 04061302 55533114 30120603 55040713 0B205361 6E746120
  4372757A 310F300D 06035504 03130620 696F7363 73301E17 0D303430 31333130
  39333132 315A170D 30373031 33303039 33313231 5A303431 0B300906 03550406
  13025553 31143012 06035504 07130B20 53616E74 61204372 757A310F 300D0603
  55040313 0620696F 73637330 819F300D 06092A86 4886F70D 01010105 0003818D
  00308189 02818100 FC0695AF 181CE90A 1B34B348 BA957178 680C8B51 07802AC3
  BF77B9C6 CB45092E 3C22292D C7D5FFC1 899185A1 FD8F37D5 C44FC206 6D1FA581
  E2264C83 1CC7453E 548C89C6 F3CD25BC 9BFFE7C5 E6653A06 62133950 78BED51B
  49128428 AB237F80 83A530EA 6F896193 F2134B54 D181F059 348AA84B 21EE6D80
  727BF668 EB004341 02030100 01A36330 61300F06 03551D13 0101FF04 05300301
  01FF300E 0603551D 0F0101FF 04040302 0186301D 0603551D 0E041604 141DA8B1
  71652961 3F7D69F0 02903AC3 2BADB137 C6301F06 03551D23 04183016 80141DA8
  B1716529 613F7D69 F002903A C32BADB1 37C6300D 06092A86 4886F70D 01010405
  00038181 00885895 A0141169 3D754EB2 E6FEC293 5BF0A80B E424AA2F A3F59765
  3463AAD1 55E71F0F B5D1A35B 9EA79DAC DDB40721 1344C01E 015BAB73 1E148E03
  9DD01431 A5E2887B 4AEC8EF4 48ACDB66 A6F9401E 8F7CA588 8A4199BB F8A437A02;
  F25064E7 112805D3 074A154F 650D09B9 8FA19347 ED359EAD 4181D9ED 0C667C10
  8A7BCFB0 FB
  quit
!
crypto provisioning registrar
 pki-server cs1
!
!
!
crypto isakmp policy 1
 hash sha
!
!
crypto ipsec transform-set test_transformset esp-aes
!
crypto map test_cryptomap 10 ipsec-isakmp
 set peer 10.23.1.10
 set security-association lifetime seconds 1800
 set transform-set test_transformset
 match address 170
!
!
interface Loopback0
 ip address 10.23.2.131 255.255.255.255
 no ip route-cache cef
 no ip route-cache
 no ip mroute-cache
!
interface FastEthernet0/0
 ip address 10.23.2.2 255.255.255.192
 no ip route-cache cef
 no ip route-cache
 no ip mroute-cache
 duplex auto
 speed auto
 crypto map test_cryptomap
!
interface FastEthernet1/0
 no ip address
 shutdown
 duplex auto
 speed auto
!
ip default-gateway 10.23.2.62
ip http server
no ip http secure-server
ip classless
ip route 0.0.0.0 0.0.0.0 10.23.2.62
!
!
access-list 170 permit ip host 10.23.2.2 host 10.23.1.10
dialer-list 1 protocol ip permit
!
!
control-plane
!
!
```

```
line con 0
 exec-timeout 0 0
 speed 115200
line aux 0
line vty 0 4
 password lab
 login
!
!
end
```

Verifying the SDP Petitioner Example

After the SDP exchange is complete, the petitioner automatically enrolls with the registrar and obtain a certificate. The following sample output through the **show running-config** command shows the automatically generated configuration, which verifies that the trustpoint is really there:

```
Router# show running-config
Building configuration...
Current configuration : 4650 bytes
!
! Last configuration change at 09:34:53 GMT Sat Jan 31 2004
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname pki-36b
!
boot-start-marker
boot-end-marker
!
logging buffered 32768 debugging
no logging console
enable secret 5 $1$JYgw$060JKXgl6dERLZpU9J3gb.
enable password lab
!
clock timezone GMT 0
no aaa new-model
ip subnet-zero
!
!
ip cef
ip domain name company.com
ip host msca-root
ip host yni-u10
ip host pki-36a 10.23.2.131
ip host pki-36a.company.com 10.23.2.131
!
!
crypto pki trustpoint tti
 enrollment url http://pki-36a.company.com:80
 revocation-check crl
 rsakeypair tti 1024
 auto-enroll 70
!
!
crypto pki certificate chain tti
 certificate 02
  308201FC 30820165 A00302012;02020102 300D0609 2A864886 F70D0101 04050030
  34310B30 09060355 04061302 55533114 30120603 55040713 0B205361 6E746120
  4372757A 310F300D 06035504 03130620 696F7363 73301E17 0D303430 31333130
  39333333 385A170D 30363031 33303039 33333338 5A302231 20301E06 092A8648
  86F70D01 09021611 706B692D 3336622E 63697363 6F2E636F 6D30819F 300D0609
  2A864886 F70D0101 01050003 818D0030 81890281 8100E383 35584B6C 24751E2C
  F4088F06 C00BFECE 84CFF8EB 50D52044 03D14A2B 91E5A260 7D07ED24 DB599D27
  432065D9 0E459248 D7CDC15D 654E2AF6 BA27D79C 23850306 3E96C508 F311D333
  76FDDC9C A810F75C FCD10F1B 9A142F0C 338B6DB3 346D3F24 97A4B15D 0A9504E7
  1F6CB769 85E9F52B FE907AAF 63D54D66 1A715A20 D7DB0203 010001A3 30302E30
  0B060355 1D0F0404 03&#048;205A0 301F0603 551D2304 18301680 141DA8B1 71652961
  3F7D69F0 02903AC3 2BADB137 C6300D06 092A8648 86F70D01 01040500 03818100
  C5E2DA0E 4312BCF8 0396014F E18B3EE9 6C970BB7 B8FAFC61 EF849568 D546F73F
  67D2A73C 156202DC 7404A394 D6124DAF 6BACB8CF 96C3141D 109C5B0E 46F4F827
  022474ED 8B59D654 F04E31A2 C9AA1152 75A0C455 FD7EEEF5 A505A648 863EE9E6
  C361D9BD E12BBB36 16B729DF 823AD5CC 404CCE48 A4379CDC 67FF6362 0601B950
  quit
 certificate ca 01
  30820241 308201AA A0030201 02020101 300D0609 2A864886 F70D0101 04050030
  34310B30 09060355 04061302 55533114 30120603 55040713 0B205361 6E746120
  4372757A 310F300D 06035504 03130620 696F7363 73301E17 0D303430 31333130
  39333132 315A170D 30373031 33303039 33313231 5A303431 0B300906 03550406
  13025553 31143012 06035504 07130B20 53616E74 61204372 757A310F 300D0603
  55040313 0620696F 73637330 819F300D 06092A86 4886F70D 01010105 0003818D
  00308189 02818100 FC0695AF 181CE90A 1B34B348 BA957178 680C8B51 07802AC3
  BF77B9C6 CB45092E 3C22292D C7D5FFC1 899185A1 FD8F37D5 C44FC206 6D1FA581
  E2264C83 1CC7453E 548C89C6 F3CD25BC 9BFFE7C5 E6653A06 62133950 78BED51B
  49128428 AB237F80 83A530EA 6F896193 F2134B54 D181F059 348AA84B 21EE6D80
  727BF668 EB004341 02030100 01A36330 61300F06 03551D13 0101FF04 05300301
  01FF300E 0603551D 0F0101FF 04040302 0186301D 0603551D 0E041604 141DA8B1
  71652961 3F7D69F0 02903AC3 2BADB137 C6301F06 03551D23 04183016 80141DA8
  B1716529 613F7D69 F002903A C32BADB1 37C6300D 06092A86 4886F70D 01010405
  00038181 00885895 A0141169 3D754EB2 E6FEC293 5BF0A80B E424AA2F A3F59765
  3463AAD1 55E71F0F B5D1A35B 9EA79DAC DDB40721 1344C01E 015BAB73 1E148E03
  9DD01431 A5E2887B 4AEC8EF4 48ACDB66 A6F9401E 8F7CA588 8A4199BB F8A437A0
  F25064E7 112805D3 074A154F 650D09B9 8FA19347 ED359EAD 4181D9ED 0C667C10
  8A7BCFB0 FB
  quit
!
no crypto engine accelerator
!
!
crypto isakmp policy 1
 hash sha
!
!
crypto ipsec transform-set test_transformset esp-aes
!
crypto map test_cryptomap 10 ipsec-isakmp
 set peer 10.23.2.2
 set security-association lifetime seconds 1800
 set transform-set test_transformset
 match address 170
!
!
interface Ethernet0/0
 ip address 10.23.1.10 255.255.255.192
 no ip route-cache cef
 no ip route-cache
 no ip mroute-cache
 half-duplex
 crypto map test_cryptomap
!
interface Ethernet0/1
 no ip address
 shutdown
 half-duplex
!
interface Ethernet0/2
 no ip address
 shutdown
 half-duplex
!
interface Ethernet0/3
 no ip address
 shutdown
 half-duplex
!
interface Serial1/0
 no ip address
 shutdown
 serial restart-delay 0
!
interface Serial1/1
 no ip address
 shutdown
 serial restart-delay 0
!
interface Serial1/2
 no ip address
 shutdown
 serial restart-delay 0
!
interface Serial1/3
 no ip address
 shutdown
 serial restart-delay 0
!
ip default-gateway 10.23.1.62
ip http server
no ip http secure-server
ip classless
ip route 0.0.0.0 0.0.0.0 10.23.1.62
!
!
access-list 170 permit ip host 10.23.1.10 host 10.23.2.2
dialer-list 1 protocol ip permit
!
!
control-plane
!
!
line con 0
 exec-timeout 0 0
 speed 115200
line aux 0
line vty 0 4
 password lab
 login
!
!
end
```

Adding AAA Lists to a RADIUS or TACACS+ Server Examples

This section contains the following configuration examples:

TACACS+ AAA Server Database Example

In the following example, user information has been added to a TACACS+ AAA database. The username is “user1.” The password is “cisco.” Two Cisco IOS configuration template variables are configured for “user1”: iosconfig1 and iosconfig2. The variables replace $1 and $2 in the configuration template file. The subject name “CN=user1, O=company, C=US” is also configured. This subject name replaces the subject name field in the subsequent enrollment request (PKCS10) that is received from the petitioner device.

```
user = user1
    password = clear "pswd"
    service=tti
        ! The certificate server inserts the following subject name to the certificate.
        set subjectname="CN=user1, O=company, C=US"
        ! Up to nine template variables may be added.
        set iosconfig1="ntp server 10.3.0.1"
        set iosconfig2="hostname user1-vpn"
```

RADIUS AAA Server Database Example

User information has been added to the RADIUS AAA server database in the following example. The username is “user1.” The password is “cisco.” Two Cisco IOS configuration template variables are configured for “user1”: iosconfig1 and iosconfig2. The variables replace $1 and $2 in the configuration template file. The subject name “CN=user1, O=company, C=US” is also configured. This subject name replaces the subject name field in the subsequent enrollment request (PKCS10) that is received from the petitioner device.

```
user = user1
    password = clear "pswd"
    radius=company
    reply_attributes=9,1="tti:subjectname=CN=user1, O=company, C=US"
    ! Up to nine template variables may be added.
    9,1="tti:iosconfig1=ntp server 10.3.0.5"
    9,1="tti:iosconfig2=hostname user1-vpn"
```

AAA List on a TACACS+ and a RADIUS AAA Server Example

The following is a configuration example showing that AAA authentication has been configured on a TACACS+ server and that AAA authorization has been configured on a RADIUS server.

Note Authentication and authorization usually point to the same server.

```
Router(config)# tacacs-server host 10.0.0.48 key cisco
Router(config)# aaa authentication login authen-tac group tacacs+
Router(config)# radius-server host 10.0.1.49 key cisco
Router(config)# aaa authorization network author-rad group radius
```

UsingaConfigurationTemplateFile Example

You can use a different configuration template file on the basis of the introducer name. For example, if you have multiple template files for different users, each with the username in the filename, configure the following under the registrar:

```
Router(config)# crypto provisioning registrar
Router (tti-registrar)# pki-server cs1
Router (tti-registrar)# template config tftp://server/config-$n.txt
```

In this example, the default configuration file shown in the section "Default Template for the Configuration File, on page 141" is used because the **template config** command does not reference a CGI script.

CGI Script Example

The following example would execute a CGI script named "mysdpcgi":

```
Router(config)# crypto provisioning registrar
Router (tti-registrar)# pki-server cs1
Router (tti-registrar)# template config tftp://server/cgi-bin/mysdpcgi post
```

The following is an example CGI script, named "mysdpcgi", that would be executed with the example **template config** command above:

```
#!/usr/bin/perl -w
# for debugging use the -debug form
# use CGI (-debug);
use CGI;
# base64 decoding is being used.
use MIME::Base64;
# The following has been commented out, but left for your information.
#
# Reading everything that has been received from stdin and writing it to the debug log to
#see what has been sent from the registrar.
#
# Remember to reset the STDIN pointer so that the normal CGI processing can get the input.
#
# print STDERR "mysdpcgi.cgi dump of stdin:\n";
# if($ENV{'REQUEST_METHOD'} eq "GET"){
#    $input_data = $ENV{'QUERY_STRING'};
# }
# else {
#    $data_length = $ENV{'CONTENT_LENGTH'};
#    $bytes_read = read(STDIN, $input_data, $data_length);
# }
# print STDERR $input_data,"\n";
# exit;

$query = new CGI;
my %av_table;
# A basic configuration file is being sent back, therefore it is being indicated as plain
# text in the command below.
print $query->header ("text/plain");
print "\n";
# For testing, parameters can be passed in so that the test applications can
# see what has been received.
#
# print STDERR "The following are the raw AV pairs mysdpcgi.cgi received:\n";
# for each $key ($query->param) {
#    print STDERR "! $key is: \n";
#    $value = $query->param($key);
#    print STDERR "! ",$value;
#    print STDERR "! \n";
#}
# The post process AV pairs are identical to those in Cisco IOS and may be used to produce
 # AV pair specific configurations as needed.
```

```
%av_table = &postprocessavpairs($query->param);
# Decoded values may be written out.
# WARNING: Some error_logs cannot handle the amount of data and will freeze.
# print STDERR "The following are the decoded AV pairs mysdpcgi.cgi received:\n";
# now write the values out
# while ( ($a, $v) = each(%av_table) ) {
#    print STDERR "$a = $v\n";
# }
# Identifying the AV pairs and specifying them in the config.
while ( ($a, $v) = each(%av_table) ) {
    if ($a eq "TTIIosRunningConfig") {
        $search = "hostname ";
        $begin = index($v, $search) + length($search);
        $end = index($v, "\n", $begin);
        $hostname = substr($v, $begin, $end - $begin);
    }
    if ($a eq "TTIIosVersion") {
        $search = "Version ";
        $begin = index($v, $search) + length($search);
        $end = index($v, "(", $begin);
        $version = substr($v, $begin, $end - $begin);
    }
}
print <<END_CONFIG;
!
! Config auto-generated by sdp.cgi
! This is for SDP testing only and is not a real config
!
!
\$t
!
\$c
!
cry pki trust Version-$version-$hostname
! NOTE: The last line of the config must be 'end' with a blank line after the end
# statement.
END_CONFIG
;
# Emulate IOS tti_postprocessavpairs functionality
sub postprocessavpairs {
    @attributes = @_;
    # Combine any AV pairs that were split apart
    $n = 0; #element index counter
    while ($attributes[$n]) {
 # see if we are at the start of a set
 if ($attributes[$n] =~ m/_0/) {
     # determine base attribute name
     $a = (split /_0/, $attributes[$n])[0];
     # set initial (partial) value
     $v = $query->param($attributes[$n]);

     # loop and pull the rest of the matching
     # attributes's values into v (would be
     # faster if we stop at first non-match)
     $c = $n+1;
     while ($attributes[$c]) {
  if ($attributes[$c] =~ m/$a/) {
      $v = $v.$query->param($attributes[$c]);
  }
  $c++;
     }

     # store in the av hash table
     $av_table{$a} = $v;
 } else {
     # store in hash table if not part of a set
     if ($attributes[$n] !~ m/_\d/) {
  $av_table{$attributes[$n]} = $query->param($attributes[$n]);
     }
 }
 $n++;
    }
    # de-base64 decode all AV pairs except userdevicename
    while ( ($a, $v) = each(%av_table) ) {
        if ($a ne "userdevicename") {
            $av_table{$a} = decode_base64($av_table{$a});
        }
    }
    return %av_table;
}
```

Note A CGI script cannot be executed without using the **post** keyword with the **template config**commandin Cisco IOS Release 12.4(6)T or a later release.

Configuring the Petitioner and Registrar for Certificate-Based Authentication Example

The following examples shows how to configure a petitioner to use the certificate issued by the trustpoint named mytrust:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# crypto provisioning petitioner

Router(tti-petitioner)# trustpoint signing mytrust

Router(tti-petitioner)# end
```

The following example shows how to configure a registrar to verify the petitioner-signing certificate and to perform authorization lookups:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# crypto provisioning registrar

Router(tti-registrar)# authentication trustpoint mytrust

Router(tti-registrar)# authorization login certificate

Router(tti-registrar)# authorization username subjectname all

Router(tti-registrar)# end
```

Configuring an Administrative Introducer Using Authentication and Authorization Lists Example

The following example shows how to configure an administrative introducer with the authentication list “authen-tac” and the authorization list “author-tac”:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# crypto provisioning registrar
Router(tti-registrar)# administrator
 authentication list authen-tac
Router(tti-registrar)# administrator
 authorization list author-tac
Router(tti-registrar)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Certificate enrollment	“ Configuring Certificate Enrollment for a PKI ” *module*
Certificate server configuration	“Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment ” module
PKI AAA integration concepts and configuration tasks	“Configuration Revocation and Authorization of Certificates in a PKI ” module
PKI commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*
USB token configuration	“ Storing PKI Credentials ” chapter in the *Cisco IOS Security Configuration Guide: Secure Connectivity* For other 12.4T features about using SDP and USB tokens to deploy PKI credentials, see the Feature Information Table.
Integrating the iPhone, iPod touch, and iPad with enterprise systems	*Apple iPhone Enterprise Deployment Guide*
Recommended cryptographic algorithms	*Next Generation Encryption*

MIBs

MIB	MIBs Link
None.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 7: Feature Information for SDP in a PKI

Feature Name	Releases	Feature Information
Secure Device Provisioning (SDP) Connect Template	12.4(20)T	This feature provides the ability to configure a device for Internet connectivity through a service provider.
USB Token and Secure Device Provisioning (SDP) Integration	12.4(15)T	This feature provides the ability to provision remote devices using a USB token as a mechanism to transfer credentials from one network device to a remote device through SDP. The following commands were introduced: **binary file**, **crypto key move rsa**, **template file**.

Feature Name	Releases	Feature Information
SDP Expanded Template CGI Support	12.4(6)T	This feature allows users to configure the SDP registrar to send a bootstrap configuration to the SDP petitioner based on not only the device name, but also its current Cisco IOS version and current configuration. The following command was modified by this feature: **template config.**
Secure Device Provisioning (SDP) Start Page	12.4(4)T	This feature allows users to configure their browsers to begin the TTI transaction by contacting the registrar's introduction URL through a start page. Thus, users no longer have to begin the TTI transaction from the welcome page on the petitioner. The following commands were introduced by this feature: **template http admin-introduction**, **template http completion**, **template http error**, **template http introduction**, **template http start**, **template http welcome.**
Administrative Secure Device Provisioning Introducer	12.3(14)T	This feature allows you to act as an administrative introducer to introduce a device into a PKI network and then provide a username as the device name for the record locator in the AAA database. The following commands were introduced by this feature: **administrator authentication list**, **administrator authorization list.**
Easy Secure Device Deployment	12.3(8)T	This feature introduces support for SDP, which offers a web-based enrollment interface that enables network administrators to deploy new devices in large networks. The following commands were introduced or modified: **crypto wui tti petitioner**, **crypto wui tti registrar**, **pki-server**, **template config**, **template username**, **trustpoint (tti-petitioner).**
Easy Secure Device Deployment AAA Integration	12.3(8)T	This feature integrates an external AAA database, allowing the SDP introducer to be authenticated against a AAA database instead of having to use the enable password of the local Cisco certificate server. The following commands were introduced or modified: **authentication list (tti-registrar)**, **authorization list (tti-registrar)**, **debug crypto wui template config**, **template username**.
Secure Device Provisioning (SDP) Certificate-Based Authorization	12.3(14)T	This feature allows certificates issued by other authority (CA) servers to be used for SDP introductions. The following commands were introduced by this feature: **administrator authentication list**, **administrator authorization list**

Feature Name	Releases	Feature Information
iPhone SDP	15.1(2)T	With the introduction of the Cisco IOS 15.1(2)T and Apple iPhone OS 3.0 releases, Apple iPhones are supported on Cisco IOS network devices. Cisco IOS routers use the SDP registrar to deploy iPhones so that network applications can be accessed securely through an IPSec VPN, SCEP server, and PKI certificate deployment technologies. The following commands were introduced by this feature: **match authentication trustpoint, match certificate** , **match url, mime-type**, **template location, template variable p, url-profile.**

CHAPTER 7

PKI Credentials Expiry Alerts

The PKI Credentials Expiry Alerts feature provides a warning mechanism in the form of an alert notification when a CA certificate is on the verge of expiry.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for PKI Credentials Expiry Alerts

Alerts are not sent for the following certificates:

- Persistent or temporary self-signed certificates.
- Secure Unique Device Identifier (SUDI) certificates.
- Certificates that belong to a trustpool. Trustpools have their own expiry alerts mechanism.
- Trustpoint clones.

Information About PKI Alerts Notification

Overview of Alerts Notification

The Cisco IOS Certificate Authority (CA) server allows autoenrollment of certificates before a certificate expires to ensure the availability of certificates for applications during authentication. However, network outages, clock update problems, and overloaded CAs can impede certificate renewal, thereby resulting in subsystems going offline because no valid certificates can be used for authentication. The PKI Credentials Expiry Alerts feature provides a mechanism by which a CA client sends a notification to a syslog server when certificates are on the verge of expiry.

The notifications are sent at the following intervals:

- First notification—This is sent 60 days before the expiry of the certificate.
- Repeated notifications—After the first notification, subsequent notifications are sent every week until a week before the expiry of the certificate. In the last week, notifications are sent every day until the certificate expiry date.

The notifications are in a *warning* mode when the certificate is valid for more than a week. The notifications are in an *alert* mode when a certificate's validity is less than a week. The notifications include the following information:

- Truspoint the certificate is associated with
- Certificate type
- Serial number of the certificate
- Certificate issuer name
- Number of days remaining for the certificate to expire
- Whether the certificate is enabled with autoenrollment
- Whether a shadow certificate is available for the corresponding certificate

Note Alert notifications are sent either via the syslog server or Simple Network Management Protocol (SNMP) traps. Notifications stop when a trustpoint is configured with autoenrollment and the corresponding shadow or rollover certificate is present, and the shadow or rollover certificate's start time is either the same or earlier than the certificate's end time.

This feature cannot be disabled and requires no additional configuration tasks. The **show crypto pki timers** command is enhanced to display the timer expiry information. The following is a sample output from the **show crypto pki timers detail** command that displays the timer when a certificate is about to expire. When this timer expires, a notification is sent to the syslog server.

```
Device# show crypto pki timers detail

PKI Timers
|       14:36.150  (2019-10-30T11:33:30Z)
```

```
  |       14:36.150  (2019-10-30T11:33:30Z) SESSION CLEANUP
  |2569d23:56:19.461  (2026-11-12T11:15:13Z) SHADOW test

Expiry Alert Timers
|659d 5:56:19.599  (2021-08-19T17:15:13Z)
  |659d 5:56:19.599  (2021-08-19T17:15:13Z) ID(test)
  |2875d 4:45:18.562  (2027-09-13T16:04:12Z) CA(test)

Trustpool Timers
|3464d 9:06:48.463  (2029-04-24T20:25:42Z)
  |3464d 9:06:48.463  (2029-04-24T20:25:42Z) TRUSTPOOL
```

The following is a syslog message that is displayed on the device:

```
Device#

Dec 16 10:24:13.533: %PKI-4-CERT_EXPIRY_WARNING: ID Certificate belonging to trustpoint tp
 will expire in 60 Days 0 hours 0 mins 0 secs.
Issuer-name cn=CA
Subject-name hostname=Router
Serial-number 02
Auto-Renewal: Not Enabled
```

PKI Traps

PKI trap ease the monitoring and operations of a PKI deployment by retrieving certificate information of the devices in the network. The root device sends SNMP traps at regular intervals to the network management system (NMS) based on the threshold configured in the device. The traps are sent in the following scenarios:

- A new certificate is installed—An SNMP trap (new certificate notification) is sent to the SNMP server containing information about the certificate, such as, certificate serial number, certificate issuer name, certificate subject name, trustpoint name, certificate type, and certificate start and end date.
- A certificate is about to expire—An SNMP trap (certificate expiry notification) is sent to the SNMP server at regular intervals starting from 60 days to one week before the certificate's end date. In the week leading up to the expiration of the certificate, the trap is sent everyday. The trap contains certificate information, such as, certificate serial number, certificate issuer name, trustpoint name, certificate type, and certificate's remaining lifetime.

To enable PKI traps, use the **snmp-server enable traps pki** command.

Note If the shadow or rollover certificate's start time is later than the certificate's end time, traps are sent stating that the shadow certificate is not yet valid. However, no traps are sent if a shadow certificate available for the same trustpoint, and the shadow certificate becomes active.

Additional References for PKI Credentials Expiry Alerts

Related Documents

Related Topic	Document Title
Cisco IOS Commands	Cisco IOS Master Command List, All Releases
Security Commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for PKI Credentials Expiry Alerts

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 8: Feature Information for PKI Credentials Expiry Alerts

Feature Name	Releases	Feature Information
PKI Credentials Expiry Alerts		The PKI Credentials Expiry Alerts feature provides a warning mechanism in the form of an alert notification when a CA certificate is on the verge of expiry. The following command was modified: **show crypto pki timers**.

CHAPTER 8

Configuring and Managing a Certificate Server for PKI Deployment

This module describes how to set up and manage a Cisco IOS certificate server for public key infrastructure (PKI) deployment. A certificate server embeds a simple certificate server, with limited certification authority (CA) functionality, into the Cisco software. Thus, the following benefits are provided to the user:

- Easier PKI deployment by defining default behavior. The user interface is simpler because default behaviors are predefined. That is, you can leverage the scaling advantages of PKI without all of the certificate extensions that a CA provides, thereby allowing you to easily enable a basic PKI-secured network.
- Direct integration with Cisco software.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

During copy, if running-config has both CA and ID certificates, if CA certificate is same as running-config, CA and ID are not replaced. Whereas, if CA certificate is different, then both ID and CA certificates gets cleared and new CA is re-inserted.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring a Certificate Server

Planning Your PKI Before Configuring the Certificate Server

Before configuring a certificate server, it is important that you have planned for and chosen appropriate values for the settings you intend to use within your PKI (such as certificate lifetimes and certificate revocation list (CRL) lifetimes). After the settings have been configured in the certificate server and certificates have been granted, settings cannot be changed without having to reconfigure the certificate server and reenrolling the peers. For information on certificate server default settings and recommended settings, see section “*Certificate Server Default Values and Recommended Values*.”

Enabling an HTTP Server

The certificate server supports Simple Certificate Enrollment Protocol (SCEP) over HTTP. The HTTP server must be enabled on the router for the certificate server to use SCEP. (To enable the HTTP server, use the **ip http server** command.) The certificate server automatically enables or disables SCEP services after the HTTP server is enabled or disabled. If the HTTP server is not enabled, only manual PKCS10 enrollment is supported.

Note To take advantage of automatic CA certificate and key pair rollover functionality for all types of certificate servers, SCEP must be used as the enrollment method.

Configuring Reliable Time Services

Time services must be running on the router because the certificate server must have reliable time knowledge. If a hardware clock is unavailable, the certificate server depends on manually configured clock settings, such as Network Time Protocol (NTP). If there is not a hardware clock or the clock is invalid, the following message is displayed at bootup:

```
% Time has not been set. Cannot start the Certificate server.
```

After the clock has been set, the certificate server automatically switches to running status.

For information on manually configuring clock settings, see the module .

Restrictions for Configuring a Certificate Server

- The certificate server does not provide a mechanism for modifying the certificate request that is received from the client; that is, the certificate that is issued from the certificate server matches the requested certificate without modifications. If a specific certificate policy, such as name constraints, must be issued, the policy must be reflected in the certificate request.
-
- For validating the HTTP connection using 3rd party open SSL, the complete ISE certificate chain is sent to the device. These certificates include the ISE certificate and its issuer CA certificate. The environment data lists these certificates.

  Cisco ISE running versions 2.7.0.310 and earlier put the certificate chain in the incoming certificate list as part of environment data. In Cisco IOS XE Release 17.1.1 and earlier releases, Cisco routers do not support multi-chain certificate downloads from ISE. Due to this, the device does not receive the ISE certificate and a TLS handshake error is displayed.

Information About Certificate Servers

RSA Key Pair and Certificate of the Certificate Server

The certificate server automatically generates a 1024-bit Rivest, Shamir, and Adelman (RSA) key pair. You must manually generate an RSA key pair if you prefer a different key pair modulus. For information on completing this task, see the section “*Generating a Certificate Server RSA Key Pair* .”

Note The recommended modulus for a certificate server RSA key pair is 2048 bits.

The certificate server uses a regular RSA key pair as its CA key. This key pair must have the same name as the certificate server. If you do not generate the key pair before the certificate server is created on the router, a general-purpose key pair is automatically generated during the configuration of the certificate server.

The CA certificate and CA key can be backed up automatically one time after they are generated by the certificate server. As a result, it is not necessary to generate an exportable CA key for backup purposes.

What to Do with Automatically Generated Key Pairs

If the key pair is automatically generated, it is not marked as exportable. Thus, you must manually generate the key pair as exportable if you want to back up the CA key. For information on how to complete this task, see the section “*Generating a Certificate Server RSA Key Pair* .”

How the CA Certificate and CA Key Are Automatically Archived

At initial certificate server setup, you can enable the CA certificate and the CA key to be automatically archived so that they may be restored later if either the original copy or the original configuration is lost.

When the certificate server is turned on the first time, the CA certificate and CA key is generated. If automatic archive is also enabled, the CA certificate and the CA key is exported (archived) to the server database. The archive can be in PKCS12 or privacy-enhanced mail (PEM) format.

Note This CA key backup file is extremely important and should be moved immediately to another secured place.

- This archiving action occurs only one time. Only the CA key that is (1) manually generated and marked exportable or (2) automatically generated by the certificate server is archived (this key is marked nonexportable).
- Autoarchiving does not occur if you generate the CA key manually and mark it “nonexportable.”
- In addition to the CA certificate and CA key archive file, you should also regularly back up the serial number file (.ser) and the CRL file (.crl). The serial file and the CRL file are both critical for CA operation if you need to restore your certificate server.
- It is not possible to manually back up a server that uses nonexportable RSA keys or manually generated, nonexportable RSA keys. Although automatically generated RSA keys are marked as nonexportable, they are automatically archived once.

Certificate Server Database

The certificate server stores files for its own use and may publish files for other processes to use. Critical files generated by the certificate server that are needed for its ongoing operation are stored to only one location per file type for its exclusive use. The certificate server reads from and writes to these files. The critical certificate server files are the serial number file (.ser) and the CRL storage location file (.crl). Files that the certificate server writes to, but does not read from again, may be published and available for use by other processes. An example of a file that may be published is the issued certificates file (.crt).

Performance of your certificate server may be affected by the following factors, which should be considered when you choose storage options and publication options for your certificate server files.

- The storage or publish locations you choose may affect your certificate server performance. Reading from a network location takes more time than reading directly from a router’s local storage device.
- The number of files you choose to store or publish to a specific location may affect your certificate server performance. The local file system may not always be suitable for a large number of files.
- The file types you choose to store or publish may affect your certificate server performance. Certain files, such as the .crl files, can become very large.

Note It is recommended that you store .ser and .crl files to your local file system and publish your .crt files to a remote file system.

Certificate Server Database File Storage

The certificate server allows the flexibility to store different critical file types to different storage locations depending on the database level set (see the **database level** command for more information). When choosing storage locations, consider the file security needed and server performance. For instance, serial number files and archive files (.p12 or .pem) might have greater security restrictions than the issued certificates file storage location (.crt) or the name file storage location (.cnm).

The table below shows the critical certificate server file types by file extension that may be stored to a specific location.

Table 9: Certificate Server Storage Critical File Types

File Extension	File Type
.ser	The main certificate server database file.
.crl	The CRL storage location.
.crt	The issued certificates storage location.
.cnm	The certificate name and expiration file storage location.
.p12	The certificate server certificate archive file location in PKCS12 format.
.pem	The certificate server certificate archive file location in PEM format.

certificate server files may be stored to three levels of specificity:

- Default location, NVRAM
- Specified primary storage location for all critical files
- Specified storage location for specific critical file(s).

A more specific storage location setting overrides a more general storage location setting. For instance, if you have not specified any certificate server file storage locations, all certificate server files are stored to NVRAM. If you specify a storage location for the name file, only the name file is stored there; all other files continue to be stored to NVRAM. If you then specify a primary location, all files except the name file is now stored to this location, instead of NVRAM.

Note You may specify either .p12 or .pem; you cannot specify both types of archive files.

Certificate Server Database File Publication

A publish file is a copy of the original file and is available for other processes to use or for your use. If the certificate server fails to publish a file, it does cause the server to shut down. You may specify one publish location for the issued certificates file and name file and multiple publish locations for the CRL file. See the table below for files types available for publication. You may publish files regardless of the database level that is set.

Table 10: Certificate Server Publish File Types

File Extension	File Type
.crl	The CRL publish location.
.crt	The issued certificates publish location.

File Extension	File Type
.cnm	The certificate name and expiration file publish location.

Trustpoint of the Certificate Server

If the certificate server also has an automatically generated trustpoint of the same name, then the trustpoint stores the certificate of the certificate server. After the router detects that a trustpoint is being used to store the certificate of the certificate server, the trustpoint is locked so that it cannot be modified.

Before configuring the certificate server you can perform the following:

- Manually create and set up this trustpoint (using the **crypto pki trustpoint**command), which allows you to specify an alternative RSA key pair (using the **rsakeypair** command).
- Specify that the initial autoenrollment key pair is generated on a specific device, such as a configured and available USB token, using the **on** command.

Note The automatically generated trustpoint and the certificate server certificate are not available for the certificate server device identity. Thus, any command-line interface (CLI) (such as the **ip http secure-trustpoint** command) that is used to specify the CA trustpoint to obtain certificates and authenticate the connecting client's certificate must point to an additional trustpoint configured on the certificate server device.

If the server is a root certificate server, it uses the RSA key pairs and several other attributes to generate a self-signed certificate. The associated CA certificate has the following key usage extensions--Digital Signature, Certificate Sign, and CRL Sign.

After the CA certificate is generated, attributes can be changed only if the certificate server is destroyed.

Note A certificate server trustpoint must not be automatically enrolled using the **auto-enroll** command. Initial enrollment of the certificate server must be initiated manually and ongoing automatic rollover functionality may be configured with the **auto-rollover** command.

Certificate Revocation Lists (CRLs)

By default, CRLs are issued once every 168 hours (1 calendar week). To specify a value other than the default value for issuing the CRL, execute the **lifetime crl** command. After the CRL is issued, it is written to the specified database location as *ca-label*.crl, where *ca-label* is the name of the certificate server.

CRLs can be distributed through SCEP, which is the default method, or a CRL distribution point (CDP), if configured and available. If you set up a CDP, use the **cdp-url** command to specify the CDP location. If the **cdp-url** command is not specified, the CDP certificate extension is not included in the certificates that are issued by the certificate server. If the CDP location is not specified, Cisco IOS PKI clients automatically request a CRL from the certificate server with a SCEP GetCRL message. The CA then returns the CRL in a SCEP CertRep message to the client. Because all SCEP messages are enveloped and signed PKCS#7 data, the SCEP retrieval of the CRL from the certificate server is costly and not highly scalable. In very large networks, an HTTP CDP provides better scalability and is recommended if you have many peer devices that check CRLs. You may specify the CDP location by a simple HTTP URL string for example,

cdp-url http://my-cdp.company.com/filename.crl

The certificate server supports only one CDP; thus, all certificates that are issued include the same CDP.

If you have PKI clients that are not running Cisco IOS software and that do not support a SCEP GetCRL request and wish to use a CDP you may set up an external server to distribute CRLs and configure the CDP to point to that server. Or, you can specify a non-SCEP request for the retrieval of the CRL from the certificate server by specifying the **cdp-url** command with the URL in the following format where *cs-addr* is the location of the certificate server:

cdp-url http://*cs-addr*/cgi-bin/pkiclient.exe?operation=GetCRL

Note If your CA is also configured as your HTTP CDP server, specify your CDP with the **cdp-url** http://*cs-addr*/cgi-bin/pkiclient.exe?operation=GetCRL command syntax.

It is the responsibility of the network administrator to ensure that the CRL is available from the location that is specified through the **cdp-url** command.

In order to force the parser to retain the embedded question mark within the specified location, enter Ctrl-v prior to the question mark. If this action is not taken, CRL retrieval through HTTP returns an error message.

The CDP location may be changed after the certificate server is running through the **cdp-url** command. New certificates contain the updated CDP location, but existing certificates are not reissued with the newly specified CDP location. When a new CRL is issued, the certificate server uses its current cached CRL to generate a new CRL. (When the certificate server is rebooted, it reloads the current CRL from the database.) A new CRL cannot be issued unless the current CRL has expired. After the current CRL expires, a new CRL is issued only after a certificate is revoked from the CLI.

Certificate Server Error Conditions

At startup, the certificate server checks the current configuration before issuing any certificates. It reports the last known error conditions through the**show crypto pki server** command output. Example errors can include any of the following conditions:

- Storage inaccessible
- Waiting for HTTP server
- Waiting for time setting

If the certificate server experiences a critical failure at any time, such as failing to publish a CRL, the certificate server automatically enters a disabled state. This state allows the network administrator to fix the condition; thereafter, the certificate server returns to the previous normal state.

Certificate Enrollment Using a Certificate Server

A certificate enrollment request functions as follows:

- The certificate server receives the enrollment request from an end user, and the following actions occur:

- A request entry is created in the enrollment request database with the initial state. (See the table below for a complete list of certificate enrollment request states.)
- The certificate server refers to the CLI configuration (or the default behavior any time a parameter is not specified) to determine the authorization of the request. Thereafter, the state of the enrollment request is updated in the enrollment request database.

- At each SCEP query for a response, the certificate server examines the current request and performs one of the following actions:
  - Responds to the end user with a “pending” or “denied” state.
  - Generates and signs the appropriate certificate and stores the certificate in the enrollment request database.

If the connection of the client has closed, the certificate server waits for the client to request another certificate.

All enrollment requests transition through the certificate enrollment states that are defined in the table below. To see current enrollment requests, use the **crypto pki server request pkcs10** command.

Table 11: Certificate Enrollment Request State Descriptions

Certificate Enrollment State	Description
authorized	The certificate server has authorized the request.
denied	The certificate server has denied the request for policy reasons.
granted	The CA core has generated the appropriate certificate for the certificate request.
initial	The request has been created by the SCEP server.
malformed	The certificate server has determined that the request is invalid for cryptographic reasons.
pending	The enrollment request must be manually accepted by the network administrator.

SCEP Enrollment

All SCEP requests are treated as new certificate enrollment requests, even if the request specifies a duplicate subject name or public key pair as a previous certificate request.

Types of CA Servers Subordinate and Registration Authorities (RAs)

CA servers have the flexibility to be configured as a subordinate certificate server or an RA-mode certificate server.

Why Configure a Subordinate CA?

A subordinate certificate server provides all the same features as a root certificate server. The root RSA key pairs are extremely important in a PKI hierarchy, and it is often advantageous to keep them offline or archived. To support this requirement, PKI hierarchies allow for subordinate CAs that have been signed by the root authority. In this way, the root authority can be kept offline (except to issue occasional CRL updates), and the subordinate CA can be used during normal operation.

Why Configure an RA-Mode Certificate Server?

A certificate server can be configured to run in RA mode. An RA offloads authentication and authorization responsibilities from a CA. When the RA receives a SCEP or manual enrollment request, the administrator can either reject or grant it on the basis of local policy. If the request is granted, it is forwarded to the issuing CA, and the CA automatically generates the certificate and return it to the RA. The client can later retrieve the granted certificate from the RA.

An RA is the authority charged with recording or verifying some or all of the data required for the CA to issue certificates. In many cases the CA undertakes all of the RA functions itself, but where a CA operates over a wide geographical area or when there is security concern over exposing the CA to direct network access, it may be administratively advisable to delegate some of the tasks to an RA and leave the CA to concentrate on its primary tasks of signing certificates and CRLs.

CA Server Compatibility

The CA server compatibility allows the IOS CA server in RA mode to interoperate with more than one type of CA server. For more information, see "*Configuring a Certificate Server to Run in RA Mode*."

Automatic CA Certificate and Key Rollover

CAs--root CAs, subordinate CAs, and RA-mode CAs--like their clients, have certificates and key pairs with expiration dates that need to be reissued when the current certificate and key pair are about to expire. When a root CA's certificate and key pair are expiring it must generate a self-signed rollover certificate and key pair. If a subordinate CA or an RA-mode CA's certificate and key pair are expiring, it requests a rollover certificate and key pair from its superior CA, obtaining the superior CA's new self-signed rollover certificates at the same time. The CA must distribute the new CA rollover certificate and keys too all its peers. This process, called rollover, allows for continuous operation of the network while the CAs and their clients are switching from an expiring CA certificate and key pair to a new CA certificate and key pair.

Rollover relies on the PKI infrastructure requirements of trust relationships and synchronized clocks. The PKI trust relationships allow (1) the new CA certificate to be authenticated, and (2) the rollover to be accomplished automatically without the loss of security. Synchronized clocks allow the rollover to be coordinated throughout your network.

Automatic CA Certificate Rollover How It Works

The CA server must have rollover configured. All levels of CAs must be automatically enrolled and have **auto-rollover** enabled. CA clients support rollover automatically when automatically enrolled. For more information about clients and automatic rollover, see the section " Automatic Certificate Enrollment " in the chapter "Configuring Certificate Enrollment for a PKI".

After CAs have rollover enabled and their clients are automatically enrolled, there are three stages to the automatic CA certificate rollover process.

Stage One: Active CA Certificate and Key Pair Only

In stage one, there is an active CA certificate and key pair only.

Stage Two: Rollover CA Certificate and Key Pair Generation and Distribution

In stage two, the rollover CA certificate and key pair are generated and distributed. The superior CA generates a rollover certificate and key pair. After the CA successfully saves its active configuration, the CA is ready to respond to client requests for the rollover certificate and key pair. When the superior CA receives a request for the new CA certificate and key pair from a client, the CA responds by sending the new rollover CA certificate and key pair to the requesting client. The clients store the rollover CA certificate and key pair.

Note When a CA generates its rollover certificate and key pair, it must be able to save its active configuration. If the current configuration has been altered, saving of the rollover certificate and key pair does not happen automatically. In this case, the administrator must save the configuration manually or rollover information is lost.

Stage Three: Rollover CA Certificate and Key Pair Become the Active CA Certificate and Key Pair

In stage three, the rollover CA certificate and key pair become the active CA certificate and key pair. All devices that have stored a valid rollover CA certificate rename the rollover certificate to the active certificate and the once-active certificate and key pair are deleted.

After the CA certificate rollover, you may observe the following deviation from usual certificate lifetime and renewal time:

- The lifetime of the certificates issued during rollover is lower than the preconfigured value.
- In specific conditions, the renew time may be inferior to the configured percentage of the actual lifetime. The difference observed can be of up to 20% in cases where the certificate lifetime is less than one hour.

These differences are normal, and result from **jitter** (random time fluctuation) introduced by the algorithm on the Certificate server. This task is performed to avoid the hosts participating to the PKI synchronize their enrollment timer, which could result in congestion on the Certificate Server.

Note The lifetime fluctuations that occur do not affect proper functionning of the PKI, since the differences always result in a shorter lifetime, thus remaining within maximum configured lifetime for certificates.

Support for Specifying a Cryptographic Hash Function

Secure Hash Algorithm (SHA) support allows a user to specify a cryptographic hash function for Cisco IOS XE certificate servers and clients. The cryptographic hash functions that can be specified are Message Digest algorithm 5 (MD5), SHA-1, SHA-256, SHA-384, or SHA-512.

Note Cisco no longer recommends using MD5; instead, you should use SHA-256 where supported. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

See the "*Configuring a Subordinate Certificate Server*" task for more information on specifying the **hash** (ca-trustpoint) and **hash** (cs-server) commands that are used to implement this feature.

How to Set Up and Deploy a Certificate Server

Generating a Certificate Server RSA Key Pair

Perform this task to manually generate an RSA key pair for the certificate server. Manually generating a certificate server RSA key pair allows you to specify the type of key pair you want to generate, to create an exportable key pair for backup purposes, to specify the key pair storage location, or to specify the key generation location.

Note You may want to create an exportable certificate server key pair for backup, or archive purposes. If this task is not performed, the certificate server automatically generates a key pair, which is not marked as exportable.

If your device has a USB token configured and available, the USB token can be used as cryptographic device in addition to a storage device. Using a USB token as a cryptographic device allows RSA operations such as key generation, signing, and authentication of credentials to be performed on a USB token. The private key never leaves the USB token and is not exportable. The public key is exportable. For titles of specific documents about configuring a USB token and making it available to use as a cryptographic device, see the “Related Documents” section.

Note It is recommended that the private key be kept in a secure location and that you regularly archive the certificate server database.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*]
4. **crypto key export rsa** *key-label* **pem** {**terminal** | **url** *url*} {**3des** | **des**} *passphrase*
5. **crypto key import rsa** *key-label* **pem** [**usage-keys** | **signature** | **encryption**] {**terminal** | **url** *url*} [**exportable**] [**on** *devicename:*] *passphrase*
6. **exit**
7. **show crypto key mypubkey rsa**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*] **Example:** `Device(config)# crypto key generate rsa label mycs exportable modulus 2048`	Generates the RSA key pair for the certificate server. • The **storage** keyword specifies the key storage location. • When specifying a label name by specifying the *key-label* argument, you must use the same name for the label that you plan to use for the certificate server (through the **crypto pki server** *cs-label*command). If a *key-label* argument is not specified, the default value, which is the fully qualified domain name (FQDN) of the router, is used. If the exportable RSA key pair is manually generated after the CA certificate has been generated, and before issuing the **no shutdown** command, then use the **crypto ca export pkcs12** command to export a PKCS12 file that contains the certificate server certificate and the private key. • By default, the modulus size of a CA RSA key is 1024 bits. The recommended modulus for a CA RSA key is 2048 bits. The range for a modulus size of a CA RSA key is from 350 to 4096 bits. • The **on** keyword specifies that the RSA key pair is created on the specified device, including a Universal Serial Bus (USB) token, local disk, or NVRAM. The name of the device is followed by a colon (:). **Note** Keys created on a USB token must be 2048 bits or less.
Step 4	**crypto key export rsa** *key-label* **pem** {**terminal** | **url** *url*} {**3des** | **des**} *passphrase* **Example:** `Device(config)# crypto key export rsa mycs pem url nvram: 3des PASSWORD`	(Optional) Exports the generated RSA key pair. Allows you to export the generated keys.

	Command or Action	Purpose
Step 5	**crypto key import rsa** *key-label* **pem** [**usage-keys** | **signature** | **encryption**] {**terminal** | **url** *url*} [**exportable**] [**on** *devicename:*] *passphrase* **Example:** `Device(config)# crypto key import rsa mycs2 pem url nvram:mycs PASSWORD`	(Optional) Imports RSA key pair. To create the imported keys on a USB token, use the **on** keyword and specify the appropriate device location. If you exported the RSA keys using the **exportable** keyword and you want to change the RSA key pair to nonexportable , import the key back to the certificate server without the **exportable** keyword. The key cannot be exported again.
Step 6	**exit** **Example:** `Device(config)# exit`	Exits global configuration.
Step 7	**show crypto key mypubkey rsa** **Example:** `Device# show crypto key mypubkey rsa`	Displays the RSA public keys of your router.

Example

The following example generates a general usage 1024-bit RSA key pair on a USB token with the label “ms2” with crypto engine debugging messages shown:

```
Device(config)# crypto key generate rsa on usbtoken0 label ms2 modulus 2048
The name for the keys will be: ms2
% The key modulus size is 2048 bits
% Generating 2048 bit RSA keys, keys will be on-token, non-exportable...
Jan 7 02:41:40.895: crypto_engine: Generate public/private keypair [OK]
Jan 7 02:44:09.623: crypto_engine: Create signature
Jan 7 02:44:10.467: crypto_engine: Verify signature
Jan 7 02:44:10.467: CryptoEngine0: CRYPTO_ISA_RSA_CREATE_PUBKEY(hw)(ipsec)
Jan 7 02:44:10.467: CryptoEngine0: CRYPTO_ISA_RSA_PUB_DECRYPT(hw)(ipsec)
```

Now, the on-token keys labeled “ms2” may be used for enrollment.

The following example shows the successful import of an encryption key to a configured and available USB tokens:

```
Device# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Device(config)# crypto key import rsa encryption on usbtoken0 url nvram:e password

% Importing public Encryption key or certificate PEM file...
filename [e-encr.pub]?
Reading file from nvram:e-encr.pub
% Importing private Encryption key PEM file...
Source filename [e-encr.prv]?
Reading file from nvram:e-encr.prv
% Key pair import succeeded.
```

Configuring Certificate Servers

Prerequisites for Automatic CA Certificate Rollover

When configuring a certificate server, for automatic CA certificate rollover to run successfully, the following prerequisites are applicable for your CA servers:

- Your CA server must be enabled and fully configured with a reliable time of day, an available key pair, a self-signed, valid CA certificate associated with the key pair, a CRL, an accessible storage device, and an active HTTP/SCEP server.
- CA clients must have successfully completed automatic enrollment and have autoenrollment enabled with the same certificate server.

Restrictions for Automatic CA Certificate Rollover

When configuring a certificate server, in order for automatic CA certificate rollover to run successfully, the following restrictions are applicable:

- SCEP must be used to support rollover. Any device that enrolls with the PKI using an alternative to SCEP as the certificate management protocol or mechanism (such as enrollment profiles, manual enrollment, or TFTP enrollment) is not be able to take advantage of the rollover functionality provided by SCEP.
- If you have automatic archive configured on your network and the archive fails, rollover does not occur because the certificate server does not enter the rollover state, and the rollover certificate and key pair is not automatically saved.

Configuring a Certificate Server

Perform this task to configure a certificate server and enable automatic rollover.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http server**
4. **crypto pki server** *cs-label*
5. **no shutdown**
6. **auto-rollover** [*time-period*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**ip http server** **Example:** `Device(config)# ip http server`	Enables the HTTP server on your system.
Step 4	**crypto pki server** *cs-label* **Example:** `Device(config)# crypto pki server server-pki`	Defines a label for the certificate server and enters certificate server configuration mode. **Note** If you manually generated an RSA key pair, the *cs-label* argument must match the name of the key pair.
Step 5	**no shutdown** **Example:** `Device(cs-server)# no shutdown`	(Optional) Enables the certificate server. **Note** Only use this command at this point if you want to use the preconfigured default functionality. That is, do not issue this command just yet if you plan to change any of the default settings as shown in the task "Configuring Certificate Server Functionality."
Step 6	**auto-rollover** [*time-period*] **Example:** `Device(cs-server)# auto-rollover 90`	(Optional) Enables the automated CA certificate rollover functionality. • *time-period*—default is 30 days.

Examples

The following example shows how to configure the certificate server "ms2" where ms2 is the label of a 2048-bit RSA key pair:

```
Device(config)# crypto pki server ms2
Device(cs-server)# no shutdown

% Once you start the server, you can no longer change some of
% the configuration.
Are you sure you want to do this? [yes/no]:
yes
% Certificate Server enabled.
Device(cs-server)# end
!
Device# show crypto pki server ms2
Certificate Server ms2:
    Status: enabled, configured
    CA cert fingerprint: 5A856122 4051347F 55E8C246 866D0AC3
    Granting mode is: manual
    Last certificate issued serial number: 0x1
    CA certificate expiration timer: 19:44:57 GMT Oct 14 2006

CRL NextUpdate timer: 19:45:25 GMT Oct 22 2003
    Current storage dir: nvram:
    Database Level: Complete - all issued certs written as <serialnum>.cer
```

The following example shows how to enable automated CA certificate rollover on the server ms2 with the **auto-rollover** command. The **show crypto pki server**command shows that the automatic rollover has been configured on the server mycs with an overlap period of 25 days.

```
Device(config)# crypto pki server ms2
Device(cs-server)# auto-rollover 25
Device(cs-server)# no shutdown
%Some server settings cannot be changed after CA certificate generation.
% Exporting Certificate Server signing certificate and keys...
% Certificate Server enabled.
Device(cs-server)#
Device# show crypto pki server ms2
Certificate Server ms2:
    Status:enabled
    Server's configuration is locked  (enter "shut" to unlock it)
    Issuer name:CN=mycs
    CA cert fingerprint:70AFECA9 211CDDCC 6AA9D7FF 3ADB03AE
    Granting mode is:manual
    Last certificate issued serial number:0x1
    CA certificate expiration timer:00:49:26 PDT Jun 20 2008
    CRL NextUpdate timer:00:49:29 PDT Jun 28 2005
    Current storage dir:nvram:
    Database Level:Minimum - no cert data written to storage
    Auto-Rollover configured, overlap period 25 days
    Autorollover timer:00:49:26 PDT May 26 2008
```

Configuring a Subordinate Certificate Server

Perform this task to configure a subordinate certificate server to grant all or certain SCEP or manual certificate requests and to enable automatic rollover.

Note Security threats, as well as the cryptographic technologies to help protect against them, are constantly changing. For more information about the latest Cisco cryptographic recommendations, see the *Next Generation Encryption* (NGE) white paper.

Before you begin

- The root certificate server should be a Cisco IOS XE certificate server.
- For a subordinate certificate authority (CA), enrollment to the root CA or upstream CA is possible only through SCEP. The upstream CA must be online for the enrollment to the upstream CA to complete. Manual enrollment of subordinate CA to the root CA or upstream CA is not possible.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**]
5. **hash** {**md5** | **sha1** | **sha256** | **sha384** | **sha512**}
6. **exit**
7. **crypto pki server** *cs-label*

8. **issuer name** *DN-string*
9. **mode sub-cs**
10. **auto-rollover** [*time-period*]
11. **grant auto rollover** {**ca-cert** | **ra-cert**}
12. **hash** {**md5** | **sha1** | **sha256** | **sha384** | **sha512**}
13. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint sub`	Declares the trustpoint that your subordinate certificate server should use and enters ca-trustpoint configuration mode.

Step 4	**enrollment** [**mode**] [**retry period** *minutes*] [**retry count** *number*] **url** *url* [**pem**]
	Example:
	`Device(ca-trustpoint)# enrollment url http://caserver.myexample.com`
	- or-
	`Device(ca-trustpoint)# enrollment url http://[2001:DB8:1:1::1]:80`
		Specifies the following enrollment parameters of the CA:
		• (Optional) The **mode** keyword specifies the registration authority (RA) mode, if your CA system provides an RA. By default, RA mode is disabled.
		• (Optional) The **retry period** keyword and *minutes* argument specifies the period, in minutes, in which the router waits before sending the CA another certificate request. Valid values are from 1 to 60. The default is 1.
		• (Optional) The **retry count** keyword and *number* argument specifies the number of times a router will resend a certificate request when it does not receive a response from the previous request. Valid values are from 1 to 100. The default is 10.
		• The *url* argument is the URL of the CA to which your router should send certificate requests.
		Note An IPv6 address can be added to the **http:** enrollment method. For example: http://[ipv6-address]:80. The IPv6 address must be enclosed in brackets in the URL. See the *enrollment url (ca-trustpoint)* command page for more information on the other enrollment methods that can be used.

	Command or Action	Purpose
		• (Optional) The **pem** keyword adds privacy-enhanced mail (PEM) boundaries to the certificate request.

Step 5	**hash** {**md5** | **sha1** | **sha256** | **sha384** | **sha512**}
	Example:
	`Device(ca-trustpoint)# hash sha384`
		(Optional) Specifies the hash function for the signature that the Cisco IOS XE client uses to sign its self-signed certificates. The Cisco IOS XE client uses the MD5 cryptographic hash function for self-signed certificates by default.
		Any of the following command algorithm keyword options can be specified to over-ride the default setting for the trustpoint. This setting then becomes the default cryptographic hash algorithm function for self-signed certificates by default.
		• **md5** —Specifies that MD5, the default hash function, is used. (No longer recommended).
		• **sha1** —Specifies that the SHA-1 hash function is used as the default hash algorithm for RSA keys. (No longer recommended).
		• **sha256** —Specifies that the SHA-256 hash function is used as the hash algorithm for Elliptic Curve (EC) 256 bit keys.
		• **sha384** —Specifies that the SHA-384 hash function is used as the hash algorithm for EC 384 bit keys.
		• **sha512** —Specifies that the SHA-512 hash function is used as the hash algorithm for EC 512 bit keys.

Step 6	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode.
Step 7	**crypto pki server** *cs-label* **Example:** `Device(config)# crypto pki server sub`	Enables a Cisco IOS XE certificate server and enters cs-server configuration mode. **Note** The subordinate server must have the same name as the trustpoint that was created in Step 3 above.
Step 8	**issuer name** *DN-string* **Example:** `Device(cs-server)# issuer-name CN=sub CA, O=Cisco, C=us`	(Optional) Specifies the DN as the CA issuer name for the certificate server.
Step 9	**mode sub-cs** **Example:** `Device(cs-server)# mode sub-cs`	Places the PKI server into sub-certificate server mode. • Sub CA and CA relationship is supported only when all the devices on the network are of Cisco IOS XE

	Command or Action	Purpose
		device type. Hence a Cisco IOS XE sub CA cannot enroll to a third party CA server.
Step 10	**auto-rollover** [*time-period*] **Example:** `Device(cs-server)# auto-rollover 90`	(Optional) Enables the automated CA certificate rollover functionality. • *time-period* --default is 30 days.

Step 11	**grant auto rollover** {**ca-cert** | **ra-cert**}
	Example:
	`Device(cs-server)# grant auto rollover ca-cert`
		(Optional) Automatically grants reenrollment requests for subordinate CAs and RA-mode CAs without operator intervention.
		• **ca-cert** --Specifies that the subordinate CA rollover certificate is automatically granted.
		• **ra-cert** --Specifies that the RA-mode CA rollover certificate is automatically granted.
		Note If this is the first time that a subordinate certificate server is enabled and enrolled, the certificate request must be manually granted.

Step 12	**hash** {**md5** | **sha1** | **sha256** | **sha384** | **sha512**}
	Example:
	`Device(cs-server)# hash sha384`
		(Optional) Sets the hash function for the signature that the Cisco IOS XE certificate authority (CA) uses to sign all of the certificates issued by the server.
		• **md5** —Specifies that MD5, the default hash function, is used. (No longer recommended).
		• **sha1** —Specifies that the SHA-1 hash function is used. (No longer recommended).
		• **sha256** —Specifies that the SHA-256 hash function is used.
		• **sha384** —Specifies that the SHA-384 hash function is used.
		• **sha512** —Specifies that the SHA-512 hash function is used.

Step 13	**no shutdown** **Example:** `Device(cs-server)# no shutdown`	Enables or reenables the certificate server. If this is the first time that a subordinate certificate server is enabled, the certificate server generates the key and obtain its signing certificate from the root certificate server.

Examples

If the certificate server fails to enable or if the certificate server has trouble handling the request that has been configured, you can use the **debug crypto pki server** command to troubleshoot your configuration as shown in the following below (Clock Not Set and Trustpoint Not Configured). Here, "ms2" refers to the label of a 2048-bit RSA key pair.

```
Router# debug crypto pki server
```

Clock Not Set

```
Router(config)# crypto pki server ms2
Router(cs-server)# mode sub-cs
Router(cs-server)# no shutdown
%Some server settings cannot be changed after CA certificate generation.
% Please enter a passphrase to protect the private key % or type Return to exit
Password:
*Jan  6 20:57:37.667: CRYPTO_CS: enter FSM: input state initial, input signal no shut
Re-enter password:
*Jan  6 20:57:45.303: CRYPTO_CS: starting enabling checks
*Jan  6 20:57:45.303: CRYPTO_CS: key 'sub' does not exist; generated automatically[OK]
% Time has not been set. Cannot start the Certificate server
```

Trustpoint Not Configured

```
Router(config)# crypto pki server ms2
Router(cs-server)# mode sub-cs
Router(cs-server)# no shutdown
%Some server settings cannot be changed after CA certificate generation.
% Please enter a passphrase to protect the private key or type Return to exit
Password:
Jan  6 21:00:15.961: CRYPTO_CS: enter FSM: input state initial, input signal no shut.
Jan  6 21:03:34.309: CRYPTO_CS: enter FSM: input state initial, input signal time set.
Jan  6 21:03:34.313: CRYPTO_CS: exit FSM: new state initial.
Jan  6 21:03:34.313: CRYPTO_CS: cs config has been unlocked
Re-enter password:
Jan  6 21:03:44.413: CRYPTO_CS: starting enabling checks
Jan  6 21:03:44.413: CRYPTO_CS: associated trust point 'sub' does not exist; generated
automatically
Jan  6 21:03:44.417: CRYPTO_CS: key 'sub' does not exist; generated automatically[OK]
Jan  6 21:04:03.993: CRYPTO_CS: nvram filesystem
Jan  6 21:04:04.077: CRYPTO_CS: serial number 0x1 written.
You must specify an enrollment URL for this CA before you can authenticate it.
% Failed to authenticate the Certificate Authority
```

If the certificate server fails to obtain its signing certificate from the root certificate server, you can use the **debug crypto pki transactions**command to troubleshoot your configuration as shown in the following example:

```
Router# debug crypto pki transactions
Jan  6 21:07:00.311: CRYPTO_CS: enter FSM: input state initial, input signal time set
Jan  6 21:07:00.311: CRYPTO_CS: exit FSM: new state initial
Jan  6 21:07:00.311: CRYPTO_CS: cs config has been unlocked no sh
%Some server settings cannot be changed after CA certificate generation.
% Please enter a passphrase to protect the private key % or type Return to exit
Password:
Jan  6 21:07:03.535: CRYPTO_CS: enter FSM: input state initial, input signal no shut
Re-enter password:
Jan  6 21:07:10.619: CRYPTO_CS: starting enabling checks
Jan  6 21:07:10.619: CRYPTO_CS: key 'sub' does not exist; generated automatically[OK]
Jan  6 21:07:20.535: %SSH-5-ENABLED: SSH 1.99 has been enabled
Jan  6 21:07:25.883: CRYPTO_CS: nvram filesystem
Jan  6 21:07:25.991: CRYPTO_CS: serial number 0x1 written.
Jan  6 21:07:27.863: CRYPTO_CS: created a new serial file.
Jan  6 21:07:27.863: CRYPTO_CS: authenticating the CA 'sub'
Jan  6 21:07:27.867: CRYPTO_PKI: Sending CA Certificate Request:
GET /cgi-bin/pkiclient.exe?operation=GetCACert&message=sub HTTP/1.0
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Cisco PKI)
Jan 6 21:07:27.867: CRYPTO_PKI: can not resolve server name/IP address
Jan 6 21:07:27.871: CRYPTO_PKI: Using unresolved IP Address 192.0.2.6 Certificate has the
following attributes:
     Fingerprint MD5: 328ACC02 52B25DB8 22F8F104 B6055B5B
     Fingerprint SHA1: 02FD799D DD40C7A8 61DC53AB 1E89A3EA 2A729EE2
% Do you accept this certificate? [yes/no]:
Jan  6 21:07:30.879: CRYPTO_PKI: http connection opened
Jan 6 21:07:30.903: CRYPTO_PKI: HTTP response header:
  HTTP/1.1 200 OK
Date: Thu, 06 Jan 2005 21:07:30 GMT
Server: server-IOS
Content-Type: application/x-x509-ca-cert
Expires: Thu, 06 Jan 2005 21:07:30 GMT
Last-Modified: Thu, 06 Jan 2005 21:07:30 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Accept-Ranges: none
Content-Type indicates we have received a CA certificate.
Jan  6 21:07:30.903: Received 507 bytes from server as CA certificate:
Jan  6 21:07:30.907: CRYPTO_PKI: transaction GetCACert completed
Jan  6 21:07:30.907: CRYPTO_PKI: CA certificate received.
Jan  6 21:07:30.907: CRYPTO_PKI: CA certificate received.
Jan  6 21:07:30.927: CRYPTO_PKI: crypto_pki_authenticate_tp_cert()
Jan  6 21:07:30.927: CRYPTO_PKI: trustpoint sub authentication status = 0 y Trustpoint CA
certificate accepted.%
% Certificate request sent to Certificate Authority
% Enrollment in progress...
Router (cs-server)#
Jan 6 21:07:51.772: CRYPTO_CA: certificate not found
Jan 6 21:07:51.772: CRYPTO_CA: certificate not found
Jan 6 21:07:52.460: CRYPTO_CS: Publishing 213 bytes to crl file nvram:sub.crl
Jan 6 21:07:54.348: CRYPTO_CS: enrolling the server's trustpoint 'sub'
Jan 6 21:07:54.352: CRYPTO_CS: exit FSM: new state check failed
Jan 6 21:07:54.352: CRYPTO_CS: cs config has been locked
Jan 6 21:07:54.356: CRYPTO_PKI: transaction PKCSReq completed
Jan 6 21:07:54.356: CRYPTO_PKI: status:
Jan 6 21:07:55.016: CRYPTO_PKI:  Certificate Request Fingerprint MD5: 1BA027DB 1C7860C7
EC188F65 64356C80
Jan 6 21:07:55.016: CRYPTO_PKI:  Certificate Request Fingerprint SHA1: 840DB52C E17614CB
0C7BE187 0DFC884D D32CAA75
Jan 6 21:07:56.508: CRYPTO_PKI: can not resolve server name/IP address
Jan 6 21:07:56.508: CRYPTO_PKI: Using unresolved IP Address 192.0.2.6
Jan 6 21:07:56.516: CRYPTO_PKI: http connection opened
Jan 6 21:07:59.136: CRYPTO_PKI: received msg of 776 bytes
Jan 6 21:07:59.136: CRYPTO_PKI: HTTP response header:
  HTTP/1.1 200 OK
  Date: Thu, 06 Jan 2005 21:07:57 GMT
  Server: server-IOS
  Content-Type: application/x-pki-message
  Expires: Thu, 06 Jan 2005 21:07:57 GMT
  Last-Modified: Thu, 06 Jan 2005 21:07:57 GMT
  Cache-Control: no-store, no-cache, must-revalidate
  Pragma: no-cache
  Accept-Ranges: none
Jan 6 21:07:59.324: The PKCS #7 message has 1 verified signers.
Jan 6 21:07:59.324: signing cert: issuer=cn=root1
Jan 6 21:07:59.324: Signed Attributes:
Jan 6 21:07:59.328: CRYPTO_PKI: status = 102: certificate request pending
Jan 6 21:08:00.788: CRYPTO_PKI: can not resolve server name/IP address
Jan 6 21:08:00.788: CRYPTO_PKI: Using unresolved IP Address 192.0.2.6
Jan 6 21:08:00.796: CRYPTO_PKI: http connection opened
Jan 6 21:08:11.804: CRYPTO_PKI: received msg of 776 bytes
Jan 6 21:08:11.804: CRYPTO_PKI: HTTP response header: HTTP/1.1 200 OK
  Date: Thu, 06 Jan 2005 21:08:01 GMT
  Server: server-IOS
  Content-Type: application/x-pki-message
  Expires: Thu, 06 Jan 2005 21:08:01 GMT
  Last-Modified: Thu, 06 Jan 2005 21:08:01 GMT
  Cache-Control: no-store, no-cache, must-revalidate
  Pragma: no-cache
  Accept-Ranges: none
Jan  6 21:08:11.992: The PKCS #7 message has 1 verified signers.
Jan  6 21:08:11.992: signing cert: issuer=cn=root1
Jan  6 21:08:11.996: Signed Attributes:
Jan  6 21:08:11.996: CRYPTO_PKI: status = 102: certificate request pending
Jan  6 21:08:21.996: CRYPTO_PKI: All sockets are closed for trustpoint sub.
Jan  6 21:08:31.996: CRYPTO_PKI: All sockets are closed for trustpoint sub.
Jan  6 21:08:41.996: CRYPTO_PKI: All sockets are closed for trustpoint sub.
Jan  6 21:08:51.996: CRYPTO_PKI: All sockets are closed for trustpoint sub.
Jan  6 21:09:01.996: CRYPTO_PKI: All sockets are closed for trustpoint sub.
Jan  6 21:09:11.996: CRYPTO_PKI: resend GetCertInitial, 1
Jan  6 21:09:11.996: CRYPTO_PKI: All sockets are closed for trustpoint sub.
Jan  6 21:09:11.996: CRYPTO_PKI: resend GetCertInitial for session: 0
Jan  6 21:09:11.996: CRYPTO_PKI: can not resolve server name/IP address
Jan  6 21:09:11.996: CRYPTO_PKI: Using unresolved IP Address 192.0.2.6
Jan  6 21:09:12.024: CRYPTO_PKI: http connection opened% Exporting Certificate Server signing
 certificate and keys...
Jan  6 21:09:14.784: CRYPTO_PKI: received msg of 1611 bytes
Jan  6 21:09:14.784: CRYPTO_PKI: HTTP response header:
  HTTP/1.1 200 OK
  Date: Thu, 06 Jan 2005 21:09:13 GMT
  Server: server-IOS
  Content-Type: application/x-pki-message
  Expires: Thu, 06 Jan 2005 21:09:13 GMT
  Last-Modified: Thu, 06 Jan 2005 21:09:13 GMT
  Cache-Control: no-store, no-cache, must-revalidate
  Pragma: no-cache
  Accept-Ranges: none
Jan  6 21:09:14.972: The PKCS #7 message has 1 verified signers.
Jan  6 21:09:14.972: signing cert: issuer=cn=root1
Jan  6 21:09:14.972: Signed Attributes:
Jan  6 21:09:14.976: CRYPTO_PKI: status = 100: certificate is granted
Jan  6 21:09:15.668: The PKCS #7 message contains 1 certs and 0 crls.
Jan  6 21:09:15.688: Newly-issued Router Cert: issuer=cn=root serial=2
Jan  6 21:09:15.688: start date: 21:08:03 GMT Jan 6 2005
Jan  6 21:09:15.688: end date: 21:08:03 GMT Jan 6 2006
Jan  6 21:09:15.688: Router date: 21:09:15 GMT Jan 6 2005
Jan  6 21:09:15.692: Received router cert from CA
Jan  6 21:09:15.740: CRYPTO_CA: certificate not found
Jan  6 21:09:15.744: CRYPTO_PKI: All enrollment requests completed for trustpoint sub.
Jan  6 21:09:15.744: %PKI-6-CERTRET: Certificate received from Certificate Authority
Jan  6 21:09:15.744: CRYPTO_PKI: All enrollment requests completed for trustpoint sub.
Jan  6 21:09:15.744: CRYPTO_PKI: All enrollment requests completed for trustpoint sub.
Jan  6 21:09:15.748: CRYPTO_CS: enter FSM: input state check failed, input signal cert
configured
Jan  6 21:09:15.748: CRYPTO_CS: starting enabling checks
Jan  6 21:09:15.748: CRYPTO_CS: nvram filesystem
Jan  6 21:09:15.796: CRYPTO_CS: found existing serial file.
Jan  6 21:09:15.820: CRYPTO_CS: old router cert flag 0x4
Jan  6 21:09:15.820: CRYPTO_CS: new router cert flag 0x44
Jan  6 21:09:18.432: CRYPTO_CS: DB version 1
Jan  6 21:09:18.432: CRYPTO_CS: last issued serial number is 0x1
Jan  6 21:09:18.480: CRYPTO_CS: CRL file sub.crl exists.
Jan  6 21:09:18.480: CRYPTO_CS: Read 213 bytes from crl file sub.crl.
Jan  6 21:09:18.532: CRYPTO_CS: SCEP server started
Jan  6 21:09:18.532: CRYPTO_CS: exit FSM: new state enabled
Jan  6 21:09:18.536: CRYPTO_CS: cs config has been locked
Jan  6 21:09:18.536: CRYPTO_PKI: All enrollment requests completed for trustpoint sub.
```

If the certificate server fails to enable or if the certificate server has trouble handling the request that has been configured, you can use the **debug crypto pki server** command to troubleshoot the progress of an enrollment. This command can also be used to debug the root CA (turn it on at the root CA).

Configuring a Certificate Server to Run in RA Mode

The certificate server can act as an RA for a CA or another third party CA. Read the details in Step 8 for more information about the **transparent** keyword option if a third-party CA is used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment url** *url*
5. **subject-name** *x.500-name*
6. **exit**
7. **crypto pki server** *cs-label*
8. **mode ra** [**transparent**]
9. **auto-rollover** [*time-period*]
10. **grant auto rollover** {**ca-cert** | **ra-cert**}
11. **no shutdown**
12. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint ra-server`	Declares the trustpoint that your RA mode certificate server should use and enters ca-trustpoint configuration mode.
Step 4	**enrollment url** *url* **Example:** `Device(ca-trustpoint)# enrollment url http://ca-server.company.com`	Specifies the enrollment URL of the issuing CA certificate server (root certificate server).

	Command or Action	Purpose
Step 5	**subject-name** *x.500-name* **Example:** `Device(ca-trustpoint)# subject-name cn=ioscs RA`	Specifies the subject name the RA uses. **Note** Include “cn=ioscs RA” or “ou=ioscs RA” in the subject name so that the issuing CA certificate server can recognize the RA (see Step 7 below).
Step 6	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode.
Step 7	**crypto pki server** *cs-label* **Example:** `Device(config)# crypto pki server ra-server`	Enables a certificate server and enters cs-server configuration mode. **Note** The certificate server must have the same name as the trustpoint that was created in Step 3 above.
Step 8	**mode ra** [**transparent**] **Example:** `Device(cs-server)# mode ra`	Places the PKI server into RA certificate server mode. Use the **transparent** keyword to allow the CA server in RA mode to interoperate with more than one type of CA server. When the **transparent** keyword is used, the original PKCS#10 enrollment message is not re-signed and is forwarded unchanged. This enrollment message makes the IOS RA certificate server work with CA servers like the Microsoft CA server.
Step 9	**auto-rollover** [*time-period*] **Example:** `Device(cs-server)# auto-rollover 90`	(Optional) Enables the automatic CA certificate rollover functionality. • *time-period* --default is 30 days.

Step 10	**grant auto rollover** {**ca-cert** | **ra-cert**}
	Example:
	`Device(cs-server)# grant auto rollover ra-cert`
		(Optional) Automatically grants reenrollment requests for subordinate CAs and RA-mode CAs without operator intervention.
		• **ca-cert** --Specifies that the subordinate CA rollover certificate is automatically granted.
		• **ra-cert** --Specifies that the RA-mode CA rollover certificate is automatically granted.
		If this is the first time that a subordinate certificate server is enabled and enrolled, the certificate request must be manually granted.

Step 11	**no shutdown** **Example:**	Enables the certificate server.

	Command or Action	Purpose
	`Device(cs-server)# no shutdown`	**Note** After this command is issued, the RA automatically enrolls with the root certificate server. After the RA certificate has been successfully received, you must issue the **no shutdown** command again, which reenables the certificate server.
Step 12	**no shutdown** **Example:** `Device(cs-server)# no shutdown`	Reenables the certificate server.

Configuring the Root Certificate Server to Delegate Enrollment Tasks to the RA Mode Certificate Server

Perform the following steps on the router that is running the issuing certificate server; that is, configure the root certificate server that is delegating enrollment tasks to the RA mode certificate server.

Note Granting enrollment requests for an RA is essentially the same process as granting enrollment requests for client devices--except that enrollment requests for an RA are displayed in the section “RA certificate requests” of the command output for the **crypto pki server info-requests** command.

SUMMARY STEPS

1. **enable**
2. **crypto pki server** *cs-label* **info requests**
3. **crypto pki server** *cs-label* **grant** *req-id*
4. **configure terminal**
5. **crypto pki server** *cs-label*
6. **grant ra-auto**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**crypto pki server** *cs-label* **info requests** **Example:** `Device# crypto pki server root-server info requests`	Displays the outstanding RA certificate request. **Note** This command is issued on the router that is running the issuing certificate server.
Step 3	**crypto pki server** *cs-label* **grant** *req-id* **Example:**	Grants the pending RA certificate request.

	Command or Action	Purpose
	`Device# crypto pki server root-server grant 9`	**Note** Because the issuing certificate server delegates the enrollment request verification task to the RA, you must pay extra attention to the RA certificate request before granting it.
Step 4	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 5	**crypto pki server** *cs-label* **Example:** `Device(config)# crypto pki server root-server`	Enables a certificate server and enters cs-server configuration mode.
Step 6	**grant ra-auto** **Example:** `Device(cs-server)# grant ra-auto`	(Optional) Specifies that all enrollment requests from an RA are to be granted automatically. **Note** For the **grant ra-auto** command to work, you have to include "cn=ioscs RA" or "ou=ioscs RA" in the subject name of the RA certificate. (See Step 2 above.)

What to Do Next

After you have configured a certificate server, you can use the preconfigured default values or specify values through the CLI for the functionality of the certificate server. If you choose to specify values other than the defaults, see the following section, "*Configuring Certificate Server Functionality* ."

Configuring Certificate Server Functionality

After you have enabled a certificate server and are in certificate server configuration mode, use any of the steps in this task to configure basic certificate server functionality values other than the default values.

Certificate Server Default Values and Recommended Values

The default values for a certificate server are intended to address a relatively small network (of about ten devices). For example, the database settings are minimal (through the **database level minimal**command) and the certificate server handles all CRL requests through SCEP. For larger networks, it is recommended that you use either the database setting "names" or "complete" (as described in the **database level** command) for possible audit and revocation purposes. Depending on the CRL checking policy, you should also use an external CDP in a larger network.

Certificate Server File Storage and Publication Locations

You have the flexibility to store file types to different storage and publication locations.

SUMMARY STEPS

1. **database url** *root-url*

2. **database url** {**cnm** | **crl** | **crt** | **p12** | **pem** | **ser**} *root-url*
3. **database url** {**cnm** | **crl** | **crt**} **publish** *root-url*
4. **database level** {**minimal** | **names** | **complete**}
5. **database username** *username* [**password** [*encr-type*] *password*]
6. **database archive** {**pkcs12** | **pem**}[**password** *encr-type*] *password*]
7. **issuer-name** *DN-string*
8. **lifetime** {**ca-certificate** | **certificate**} *time*
9. **lifetime crl** *time*
10. **lifetime enrollment-request** *time*
11. **cdp-url** *url*
12. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**database url** *root-url* **Example:** `Device(cs-server)# database url tftp://cert-svr-db.company.com`	Specifies the primary location where database entries for the certificate server are written. If this command is not specified, all database entries are written to NVRAM.
Step 2	**database url** {**cnm** | **crl** | **crt** | **p12** | **pem** | **ser**} *root-url* **Example:** `Device(cs-server)# database url ser nvram:`	Specifies certificate server critical file storage location by file type. **Note** If this command is not specified, all critical files are stored to the primary location if specified. If the primary location is not specified, all critical files are stored to NVRAM.
Step 3	**database url** {**cnm** | **crl** | **crt**} **publish** *root-url* **Example:** `Device(cs-server)# database url crl publish tftp://csdb_specific_crl_files.company.com`	Specifies certificate server publish location by file type. **Note** If this command is not specified, all publish files are stored to the primary location if specified. If the primary location is not specified, all publish files are stored to NVRAM.

Step 4	**database level** {**minimal** | **names** | **complete**}
	Example:
	`Device(cs-server)# database level complete`
		Controls what type of data is stored in the certificate enrollment database.
		• **minimal** --Enough information is stored only to continue issuing new certificates without conflict; the default value.
		• **names** --In addition to the information given in the minimal level, the serial number and subject name of each certificate.
		• **complete** --In addition to the information given in the minimal and names levels, each issued certificate is written to the database.

	Command or Action	Purpose
		Note The **complete** keyword produces a large amount of information; if it is issued, you should also specify an external TFTP server in which to store the data through the **database url** command.
Step 5	**database username** *username* [**password** [*encr-type*] *password*] **Example:** `Device(cs-server)# database username user password PASSWORD`	(Optional) Sets a username and password when a user is required to access a primary certificate enrollment database storage location.
Step 6	**database archive** {**pkcs12** | **pem**}[**password** *encr-type*] *password*] **Example:** `Device(cs-server)# database archive pem`	(Optional) Sets the CA key and CA certificate archive format and password to encrypt the file. The default value is **pkcs12**, so if this subcommand is not configured, autoarchiving continues, and the PKCS12 format is used. • The password is optional. If it is not configured, you are prompted for the password when the server is turned on for the first time. **Note** It is recommended that you remove the password from the configuration after the archive is finished.
Step 7	**issuer-name** *DN-string* **Example:** `Device(cs-server)# issuer-name my-server`	(Optional) Sets the CA issuer name to the specified distinguished name (*DN-string*). The default value is as follows: **issuer-name cn**={*cs-label* }.
Step 8	**lifetime** {**ca-certificate** | **certificate**} *time* **Example:** `Device(cs-server)# lifetime certificate 888`	(Optional) Specifies the lifetime, in days, of a CA certificate or a certificate. Valid values range from 1 day to 1825 days. The default CA certificate lifetime is 3 years; the default certificate lifetime is 1 year. The maximum certificate lifetime is 1 month less than the lifetime of the CA certificate.
Step 9	**lifetime crl** *time* **Example:** `Device(cs-server)# lifetime crl 333`	(Optional) Defines the lifetime, in hours, of the CRL that is used by the certificate server. Maximum lifetime value is 336 hours (2 weeks). The default value is 168 hours (1 week).
Step 10	**lifetime enrollment-request** *time* **Example:** `Device(cs-server)# lifetime enrollment-request 888`	(Optional) Specifies how long an enrollment request should stay in the enrollment database before being removed. Maximum lifetime is 1000 hours.

	Command or Action	Purpose
Step 11	**cdp-url** *url* **Example:** `Device(cs-server)# cdp-url http://my-cdp.company.com`	(Optional) Defines the CDP location to be used in the certificates that are issued by the certificate server. • The URL must be an HTTP URL. If you have PKI clients that are not running Cisco IOS software and that do not support a SCEP GetCRL request, use the following URL format: http://server.company.com/certEnroll/filename.crl Or, if your Cisco IOS certificate server is also configured as your CDP, use the following URL format http://cs-addr/cgi-bin/pkiclient.exe?operation=GetCRL where *cs-addr* is the location of the certificate server. In order to force the parser to retain the embedded question mark within the specified location, enter Ctrl-v prior to the question mark. If this action is not taken, CRL retrieval through HTTP returns an error message. **Note** Although this command is optional, it is strongly recommended for any deployment scenario.
Step 12	**no shutdown** **Example:** `Device(cs-server)# no shutdown`	Enables the certificate server. You should issue this command only after you have completely configured your certificate server.

Examples

The following example shows how to configure a CDP location where the PKI clients do not support SCEP GetCRL requests:

```
Device(config)# crypto pki server aaa
Device(cs-server)# database level minimum
Device(cs-server)# database url tftp://10.1.1.1/username1/
Device(cs-server)# issuer-name CN=aaa
Device(cs-server)# cdp-url http://server.company.com/certEnroll/aaa.crl
```

After a certificate server has been enabled on a router, the **show crypto pki server**command displays the following output:

```
Device# show crypto pki server

    Certificate Server status:enabled, configured
    Granting mode is:manual
    Last certificate issued serial number:0x1
    CA certificate expiration timer:19:31:15 PST Nov 17 2006
    CRL NextUpdate timer:19:31:15 PST Nov 25 2003
    Current storage dir:nvram:
    Database Level:Minimum - no cert data written to storage
```

Working with Automatic CA Certificate Rollover

Starting Automated CA Certificate Rollover Immediately

Use this task to initiate the automated CA certificate rollover process immediately on your root CA server.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki server** *cs-label* **rollover** [**cancel**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki server** *cs-label* **rollover** [**cancel**] **Example:** `Device(config)# crypto pki server mycs rollover`	Immediately starts the CA certificate rollover process by generating a shadow CA certificate. To delete the CA certificate rollover certificate and keys, use the **cancel** keyword.

Requesting a Certificate Server Client Rollover Certificate

Use this task to request a certificate server client's rollover certificate.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki server** *cs-label* **rollover request pkcs10 terminal**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**crypto pki server** *cs-label* **rollover request pkcs10 terminal** **Example:** `Device(config)# crypto pki server mycs rollover request pkcs10 terminal`	Requests a client rollover certificate from the server.

Example

The following example shows a rollover certificate request being inputted into the server:

```
Device# crypto pki server mycs rollover request pkcs10 terminal

% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIBUTCBuwIBADASMRAwDgYDVQQDEwdOZXdSb290MIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDMHeev1ERSs320zbLQQk+3lhV/R2HpYQ/iM6uT1jkJf5iy0UPR
wF/X16yUNmG+ObiGiW9fsASF0nxZw+fO7d2X2yh1PakfvF2wbP27C/sgJNOw9uPf
sBxEc40Xe0d5FMh0YKOSAShfZYKOflnyQR2Drmm2x/33QGol5QyRvjkeWQIDAQAB
oAAwDQYJKoZIhvcNAQEEBQADgYEALM90r4d79X6vxhD0qjuYJXfBCOvv4FNyFsjr
aBS/y6CnNVYySF8UBUohXYIGTWf4I4+sj6i8gYfoFUW1/L82djS18TLrUr6wpCOs
RqfAfps7HW1e4cizOfjAUU+C7lNcobCAhwF1o6q2nIEjpQ/2yfK9O7sb3SCJZBfe
eW3tyCo=
-----END CERTIFICATE REQUEST-----
```

Exporting a CA Rollover Certificate

Use this task to export a CA rollover certificate.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki export** *trustpoint* **pem** {**terminal** | **url** *url*} [**rollover**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki export** *trustpoint* **pem** {**terminal** | **url** *url*} [**rollover**]	Exports a CA shadow certificate.

	Command or Action	Purpose
	Example: `Device(config)# crypto pki export mycs pem terminal rollover`	

Maintaining Verifying and Troubleshooting the Certificate Server Certificates and the CA

Managing the Enrollment Request Database

SCEP supports two client authentication mechanisms--manual and preshared key. Manual enrollment requires the administrator at the CA server to specifically authorize the enrollment requests; enrollment using preshared keys allows the administrator to preauthorize enrollment requests by generating a one-time password (OTP).

Use any of the optional steps within this task to help manage the enrollment request database by performing functions such as specifying enrollment processing parameters that are to be used by SCEP and by controlling the run-time behavior or the certificate server.

SUMMARY STEPS

1. **enable**
2. **crypto pki server** *cs-label* **grant** {**all** | *req-id*}
3. **crypto pki server** *cs-label* **reject** {**all** | *req-id*}
4. **crypto pki server** *cs-label* **password generate** *minutes*
5. **crypto pki server** *cs-label* **revoke** *certificate-serial-number*
6. **crypto pki server** *cs-label* **request pkcs10** {*url* | **terminal**} [**base64**| **pem**
7. **show crypto pki server** *cs-label* **crl**
8. **show crypto pki server** *cs-label* **requests**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Step 2	**crypto pki server** *cs-label* **grant** {**all** | *req-id*}
	Example:
	`Device# crypto pki server mycs grant all`
		Grants all or specific SCEP requests.

Step 3	**crypto pki server** *cs-label* **reject** {**all** | *req-id*}
	Example:
	`Device# crypto pki server mycs reject all`
		Rejects all or specific SCEP requests.

Step 4	**crypto pki server** *cs-label* **password generate** *minutes* **Example:**	Generates a OTP for SCEP requests.

	Command or Action	Purpose
	`Device# crypto pki server mycs password generate 75`	• *minutes* --Length of time, in minutes, that the password is valid. Valid values range from 1 to 1440 minutes. The default is 60 minutes. **Note** Only one OTP is valid at a time; if a second OTP is generated, the previous OTP is no longer valid.
Step 5	**crypto pki server** *cs-label* **revoke** *certificate-serial-number* **Example:** `Device# crypto pki server mycs revoke 3`	Revokes a certificate on the basis of its serial number. • *certificate-serial-number* --One of the following options: • A string with a leading 0x, which is treated as a hexadecimal value • A string with a leading 0 and no x, which is treated as octal • All other strings, which are treated as decimal
Step 6	**crypto pki server** *cs-label* **request pkcs10** {*url* | **terminal**} [**base64**| **pem** **Example:** `Device# crypto pki server mycs request pkcs10 terminal pem`	Manually adds either a base64-encoded or PEM-formatted PKCS10 certificate enrollment request to the request database. After the certificate is granted, it is displayed on the console terminal using base64 encoding. • **pem** --Specifies the certificate that is returned with PEM headers automatically added to the certificate after the certificate is granted, regardless of whether PEM headers were used in the request. • **base64** --Specifies the certificate that is returned without privacy-enhanced mail (PEM) headers, regardless of whether PEM headers were used in the request.
Step 7	**show crypto pki server** *cs-label* **crl** **Example:** `Device# show crypto pki server mycs crl`	Displays information regarding the status of the current CRL.
Step 8	**show crypto pki server** *cs-label* **requests** **Example:** `Device# show crypto pki server mycs requests`	Displays all outstanding certificate enrollment requests.

Removing Requests from the Enrollment Request Database

After the certificate server receives an enrollment request, the server can leave the request in a pending state, reject it, or grant it. The request stays in the enrollment request database for 1 week until the client polls the certificate server for the result of the request. If the client exits and never polls the certificate server, you can remove either individual requests or all requests from the database.

Use this task to remove requests from the database and allow the server to be returned to a clean slate with respect to the keys and transaction IDs. Also, you can use this task to help troubleshoot a SCEP client that may not be behaving properly.

SUMMARY STEPS

1. **enable**
2. **crypto pki server** *cs-label* **remove** {**all** | *req-id*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Step 2	**crypto pki server** *cs-label* **remove** {**all** | *req-id*}
	Example:
	`Device# crypto pki server mycs remove 15`
		Removes enrollment requests from the enrollment request database.

Deleting a Certificate Server

Users can delete a certificate server from the PKI configuration if they no longer want it on the configuration. Typically, a subordinate certificate server or an RA is being deleted. However, users may delete a root certificate server if they are moving it to another device through the archived RSA keys.

Perform this task to delete a certificate server from your PKI configuration.

Note When a certificate server is deleted, the associated trustpoint and key are also deleted.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no crypto pki server** *cs-label*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**no crypto pki server** *cs-label* **Example:** `Device(config)# no crypto pki server mycs`	Deletes a certificate server and associated trustpoint and key.

Verifying and Troubleshooting Certificate Server and CA Status

Use any of the following optional steps to verify the status of the certificate server or the CA.

SUMMARY STEPS

1. **enable**
2. **debug crypto pki server**
3. **dir** *filesystem* :

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**debug crypto pki server** **Example:** `Device# debug crypto pki server`	Enables debugging for a crypto PKI certificate server. - This command can be used for monitoring the progress of an enrollment and for troubleshooting if the certificate server fails to respond or if the certificate server has trouble handling the request that has been configured.
Step 3	**dir** *filesystem* : **Example:** `Device# dir slot0:`	Displays a list of files on a file system. - This command can be used to verify the certificate server autoarchived file if the **database url** command was entered to point to a local file system. You should be able to at least see “*cs-label* .ser” and “*cs-label* .crl” files in the database.

Verifying CA Certificate Information

To obtain information relating to the CA certificates including the certificate server rollover process, rollover certificates, and timers, you may use any of the following commands.

Note These commands are not exclusive to shadow certificate information. If no shadow certificate exists, the following commands display the active certificate information.

SUMMARY STEPS

1. **crypto pki certificate chain**
2. **crypto pki server info requests**
3. **show crypto pki certificates**
4. **show crypto pki server**
5. **show crypto pki trustpoints**

DETAILED STEPS

Step 1 **crypto pki certificate chain**

Example:

```
Device(config)# crypto pki certificate chain mica

certificate 06
certificate ca 01
! This is the peer’s shadow PKI certificate.
certificate rollover 0B
! This is the CA shadow PKI certificate
certificate rollover ca 0A
```

Displays the certificate chain details and to distinguish the current active certificate from the rollover certificate in the certificate chain. The following example shows a certificate chain with an active CA certificate and a shadow, or rollover, certificate:

Step 2 **crypto pki server info requests**

Example:

```
Device# crypto pki server myca info requests

Enrollment Request Database:
RA certificate requests:
  ReqID  State      Fingerprint                      SubjectName
--------------------------------------------------------------
RA rollover certificate requests:
  ReqID  State      Fingerprint                      SubjectName
  --------------------------------------------------------------
Router certificates requests:
  ReqID  State      Fingerprint                      SubjectName
--------------------------------------------------------------
1      pending    A426AF07FE3A4BB69062E0E47198E5BF hostname=client
  Router rollover certificates requests:
  ReqID  State      Fingerprint                      SubjectName
  --------------------------------------------------------------
  2      pending    B69062E0E47198E5BFA426AF07FE3A4B hostname=client
```

Displays all outstanding certificate enrollment requests. The following example shows the output for shadow PKI certificate information requests:

Step 3 **show crypto pki certificates**

Example:

```
Device# show crypto pki certificates

Certificate
  Subject Name
    Name: myrouter.example.com
    IP Address: 192.0.2.1
    Serial Number: 04806682
  Status: Pending
  Key Usage: General Purpose
    Fingerprint: 428125BD A3419600 3F6C7831 6CD8FA95 00000000
CA Certificate
  Status: Available
  Certificate Serial Number: 3051DF7123BEE31B8341DFE4B3A338E5F
  Key Usage: Not Set
```

Displays information about the certificate, the certification authority certificate, shadow certificates, and any registration authority certificates. The following example displays the certificate of the router and the certificate of the CA. There is no shadow certificate available. A single, general-purpose RSA key pair was previously generated, and a certificate was requested but not received for that key pair. Note that the certificate status of the router shows "Pending." After the router receives its certificate from the CA, the Status field changes to "Available" in the **show** output.

Step 4 **show crypto pki server**

Example:

```
Device# show crypto pki server

Certificate Server routercs:
   Status: enabled, configured
   Issuer name: CN=walnutcs
   CA cert fingerprint: 800F5944 74337E5B C2DF6C52 9A7B1BDB
   Granting mode is: auto
   Last certificate issued serial number: 0x7
   CA certificate expiration timer: 22:10:29 GMT Jan 29 2007
   CRL NextUpdate timer: 21:50:56 GMT Mar 5 2004
   Current storage dir: nvram:
   Database Level: Minimum - no cert data written to storage
Rollover status: available for rollover
    Rollover CA cert fingerprint: 6AAF5944 74227A5B 23DF3E52 9A7F1FEF
    Rollover CA certificate expiration timer: 22:10:29 GMT Jan 29 2017
```

Displays the current state and configuration of the certificate server. The following example shows that the certificate server "routercs" has rollover configured. The CA auto-rollover time has occurred and the rollover, or shadow, PKI certificate is available. The status shows the rollover certificate fingerprint and rollover CA certificate expiration timer information.

Step 5 **show crypto pki trustpoints**

Example:

```
Device# show crypto pki trustpoints

Trustpoint vpn:
   Subject Name:
   cn=Cisco SSL CA
   o=Cisco Systems
   Serial Number: 0FFEBBDC1B6F6D9D0EA7875875E4C695
   Certificate configured.
   Rollover certificate configured.
   Enrollment Protocol:
   SCEPv1, PKI Rollover
```

Displays the trustpoints that are configured in the device. The following output shows that a shadow CA certificate is available and shows the SCEP capabilities reported during the last enrollment operation:

Configuration Examples for Using a Certificate Server

Example: Configuring Specific Storage and Publication Locations

The following example shows the configuration of a minimal local file system, so that the certificate server can respond quickly to certificate requests. The .ser and .crl files are stored on the local system for fast access, and a copy of all of the .crt files are published to a remote location for long-term logging.

```
crypto pki server myserver
     !Pick your database level.
     database level minimum
     !Specify a location for the .crt files that is different than the default local
!Cisco IOS file system.
     database url crt publish http://url username user1 password secret
```

Note Free space on the local file system should be monitored, in case the .crl file becomes too large.

The following example shows the configuration of a primary storage location for critical files, a specific storage location for the critical file serial number file, the main certificate server database file, and a password protected file publication location for the CRL file:

```
Device(config)# crypto pki server mycs
Device(cs-server)# database url ftp://cs-db.company.com

!
% Server database url was changed. You need to move the
% existing database to the new location.
!
Device(cs-server)# database url ser nvram:
Device(cs-server)# database url crl publish ftp://crl.company.com username myname password
 mypassword
Device(cs-server)# end
```

The following output displays the specified primary storage location and critical file storage locations specified:

```
Device# show

Sep  3 20:19:34.216: %SYS-5-CONFIG_I: Configured from console by user on console
Device# show crypto pki server

Certificate Server mycs:
     Status: disabled
     Server's configuration is unlocked  (enter "no shut" to lock it)
     Issuer name: CN=mycs
     CA cert fingerprint: -Not found-
     Granting mode is: manual
     Last certificate issued serial number: 0x0
     CA certificate expiration timer: 00:00:00 GMT Jan 1 1970
     CRL not present.
     Current primary storage dir: ftp://cs-db.company.com
     Current storage dir for .ser files: nvram:
    Database Level: Minimum - no cert data written to storage The following output displays
 all storage and publication locations. The serial number file (.ser) is stored in NVRAM.
```

```
The CRL file will be published to ftp://crl.company.com with a username and password. All
other critical files will be stored to the primary location, ftp://cs-db.company.com.

Device# show running-config

   section crypto pki server
   crypto pki server mycs shutdown database url ftp://cs-db.company.com
   database url crl publish ftp://crl.company.com username myname password 7
12141C0713181F13253920
   database url ser nvram:
Device#
```

Example: Removing Enrollment Requests from the Enrollment Request Database

The following examples show both the enrollment requests that are currently in the enrollment request database and the result after one of the enrollment requests has been removed from the database.

Example: Enrollment Request Currently in the Enrollment Request Database

The following example shows that the **crypto pki server info requests** command has been used to display the enrollment requests that are currently in the Enrollment Request Database:

```
Device# crypto pki server myserver info requests

Enrollment Request Database:
RA certificate requests:
ReqID    State      Fingerprint                               SubjectName
------------------------------------------------------------------------
Router certificates requests:
ReqID    State      Fingerprint                               SubjectName
------------------------------------------------------------------------
2        pending    1B07F3021DAAB0F19F35DA25D01D8567      hostname=host1.company.com
1        denied     5322459D2DC70B3F8EF3D03A795CF636      hostname=host2.company.com
```

Example: crypto pki server remove Command Used to Remove One Enrollment Request

The following example shows that the **crypto pki server remove** command has been used to remove Enrollment Request 1:

```
Device# crypto pki server myserver remove 1
```

Example: Enrollment Request Database After the Removal of One Enrollment Request

The following example shows the result of the removal of Enrollment Request 1 from the Enrollment Request Database:

```
Device# crypto pki server mycs info requests

Enrollment Request Database:
RA certificate requests:
ReqID    State     Fingerprint                        SubjectName
-----------------------------------------------------------------
Router certificates requests:
ReqID    State     Fingerprint                        SubjectName
-----------------------------------------------------------------
2        pending   1B07F3021DAAB0F19F35DA25D01D8567   hostname=host1.company.com
```

Example: Autoarchiving the Certificate Server Root Keys

The following output configurations and examples show what you might see if the **database archive** command has not been configured (that is, configured using the default value); if the **database archive** command has been configured to set the CA certificate and CA key archive format as PEM, without configuring a password; and if the **database archive** command has been configured to set the CA certificate and CA key archive format as PKCS12, with a password configured. The last example is sample content of a PEM-formatted archive file. The following example, “ms2” refers to the label of a 2048-bit key pair.

Example: database archive Command Not Configured

Note The default is PKCS12, and the prompt for the password appears after the **no shutdown** command has been issued.

```
Device(config)# crypto pki server ms2
Device(cs-server)# no shutdown

% Ready to generate the CA certificate.
%Some server settings cannot be changed after CA certificate generation.
Are you sure you want to do this? [yes/no]: y
% Exporting Certificate Server signing certificate and keys...
! Note the next two lines, which are asking for a password.
% Please enter a passphrase to protect the private key.
Password:
% Certificate Server enabled.
Device(cs-server)# end
Device# dir nvram:

Directory of nvram:/
  125  -rw-        1693               <no date>  startup-config
  126  ----           5               <no date>  private-config
    1  -rw-          32               <no date>  myserver.ser
    2  -rw-         214               <no date>  myserver.crl
! Note the next line, which indicates PKCS12 format.
    3  -rw-        1499               <no date>  myserver.p12
```

Example" database archive Command and pem Keyword Configured

Note The prompt for the password appears after the **no shutdown** command has been issued.

```
Device(config)# crypto pki server ms2
Device(cs-server)# database archive pem
Device(cs-server)# no shutdown

% Ready to generate the CA certificate.
%Some server settings cannot be changed after CA certificate generation.
Are you sure you want to do this? [yes/no]: y
% Exporting Certificate Server signing certificate and keys...
!Note the next two lines, which are asking for a password.
% Please enter a passphrase to protect the private key.
Password:
% Certificate Server enabled.
```

```
Device(cs-server)# end
Device# dir nvram

Directory of nvram:/
  125  -rw-        1693            <no date>  startup-config
  126  ----           5            <no date>  private-config
    1  -rw-          32            <no date>  myserver.ser
    2  -rw-         214            <no date>  myserver.crl
! Note the next line showing that the format is PEM.
    3  -rw-        1705            <no date>  myserver.pem
```

Example: database archive Command and pkcs12 Keyword (and Password) Configured

Note When the password is entered, it is encrypted. However, it is recommended that you remove the password from the configuration after the archive has finished.

```
Device(config)# crypto pki server ms2
Device(cs-server)# database archive pkcs12 password cisco123
Device(cs-server)# no shutdown

% Ready to generate the CA certificate.
% Some server settings cannot be changed after CA certificate generation.
Are you sure you want to do this? [yes/no]: y
% Exporting Certificate Server signing certificate and keys...
! Note that you are not being prompted for a password.
% Certificate Server enabled.
Device(cs-server)# end
Device# dir nvram:

Directory of nvram:/
   125   -rw-        1693            <no date>   startup-config
   126   ----           5            <no date>   private-config
     1   -rw-          32            <no date>   myserver.ser
     2   -rw-         214            <no date>   myserver.crl
! Note that the next line indicates that the format is PKCS12.
     3   -rw-        1499            <no date>   myserver.p12
```

Example: PEM-Formatted Archive

The following sample output shows that autoarchiving has been configured in PEM file format. The archive consists of the CA certificate and the CA private key. To restore the certificate server using the backup, you would have to import the PEM-formatted CA certificate and CA key individually.

Note In addition to the CA certificate and CA key archive files, you should also back up the serial file (.ser) and the CRL file (.crl) regularly. The serial file and the CRL file are both critical for CA operation if you need to restore your certificate server.

```
Device# more nvram:mycs.pem

-----BEGIN CERTIFICATE-----
MIIB9zCCAWCgAwIBAgIBATANBgkqhkiG9w0BAQQFADAPMQ0wCwYDVQQDEwRteWNz
MB4XDTA0MDgyNzAyMzI0NloXDTA3MDgyNzAyMzI0NlowDzENMAsGA1UEAxMEbXlj
czCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA1lZpKP4nGDJHgPkpYSkix7lD
```

```
nr23aMlZ9Kz5oo/qTBxeZ8mujpjYcZ0T8AZvoOiCuDnYMl796ZwpkMgjz1aZZbL+
BtuVvllsEOfhC+u/Ol/vxfGG5xpshoz/F5J3xdg5ZZuWWuIDAUYu9+QbI5feuG04
Z/BiPIb4AmGTP4B2MM0CAwEAAaNjMGEwDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8B
Af8EBAMCAYYwHwYDVR0jBBgwFoAUKi/cuK6wkz+ZswVtb06vUJboEeEwHQYDVR0O
BBYEFCov3LiusJM/mbMFbW9Or1CW6BHhMA0GCSqGSIb3DQEBBAUAA4GBAKLOmoE2
4+NeOKEXMCXG1jcohK7O2HrkFfl/vpK0+q92PTnMUFhxLOqI8pWIq5CCgC7heace
OrTv2zcUAoH4rzx3Rc2USIxkDokWWQMLujsMm/SLIeHit0G5uj//GCcbgK20MAW6
ymf7+TmblSFljWzstoUXC2hLnsJIMq/KffaD
-----END CERTIFICATE-----

!The private key is protected by the password that is
configured in “database archive pem password pwd” or that
is entered when you are prompted for the password.
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,106CE91FFD0A075E

zyiFC8rKv8Cs+IKsQG2QpsVpvDBHqZqBSM4D528bvZv7jzr6WuHj8E6zO+6G8R/A
zjsfTALo+e+ZDg7KMzbryHARvjskbqFdOMLlVIYBhCeSElKsskWB6chOuyPHJInW
JwC5YzZdZwOqcyLBP/xOYXcvjzzNfPAXZzN12VR8vWDNq/kHT+3Lplc8hY++ABMI
M+C9FB3dpNZzu5O1BZCJg46bqbkulaCCmScIDaVt0zDFZwWTSufiemmNxZBG4xS8
t5t+FEhmSfv8DAmwg4f/KVRFTm10phUArcLxQO38Al0W5YHHORdACnuzVUvHgco7
VT4XUTjO7qMhmJgFNWy1pu49fbdS2NnOn5IoiyAq5lk1KUPrz/WABWiCvLMylGnZ
kyMCWoaMtgS/vdx74BBCj09yRZJnLMlIi6SDofjCNTDHfmFEVg4LsSWCd4lP9OP8
0MqhP1D5VIx6PbMNwkWW12lpBbCCdesFRGHjZD2dOu96kHD7ItErx34CC8W04aG4
b7DLktUu6WNV6M8g3CAqJiC0V8ATlp+kvdHZVkXovgND5IU0OJpsj0HhGzKAGpOY
KTGTUekUboISjVVkI6efp1vO6temVL3Txg3KGhzWMJGrq1snghE0KnV8tkddv/9N
d/t1l+we9mrccTq50WNDnkEi/cwHI/0PKXg+NDNH3k3QGpAprsqGQmMPdqc5ut0P
86i4cF9078QwWg4Tpay3uqNH1Zz6UN0tcarVVNmDupFESUxYw10qJrrEYVRadu74
rKAU4Ey4xkAftB2kuqvr21Av/L+jne4kkGIoZYdB+p/M98pQRgkYyg==
-----END RSA PRIVATE KEY-----
```

Example: Restoring a Certificate Server from Certificate Server Backup Files

The following example shows that restoration is from a PKCS12 archive and that the database URL is NVRAM (the default).

```
Device# copy tftp://192.0.2.71/backup.ser nvram:mycs.ser

Destination filename [mycs.ser]?
32 bytes copied in 1.320 secs (24 bytes/sec)

Device# copy tftp://192.0.2.71/backup.crl nvram:mycs.crl

Destination filename [mycs.crl]?
214 bytes copied in 1.324 secs (162 bytes/sec)

Device# configure terminal
Device(config)# crypto pki import mycs pkcs12 tftp://192.0.2.71/backup.p12 cisco123

Source filename [backup.p12]?
CRYPTO_PKI: Imported PKCS12 file successfully.

Device(config)# crypto pki server mycs
! fill in any certificate server configuration here

Device(cs-server)# no shutdown
% Certificate Server enabled.

Device(cs-server)# end
Device# show crypto pki server
```

```
Certificate Server mycs:
    Status: enabled
    Server's current state: enabled
    Issuer name: CN=mycs
    CA cert fingerprint: 34885330 B13EAD45 196DA461 B43E813F
    Granting mode is: manual
    Last certificate issued serial number: 0x1
    CA certificate expiration timer: 01:49:13 GMT Aug 28 2007
    CRL NextUpdate timer: 01:49:16 GMT Sep 4 2004
    Current storage dir: nvram:
    Database Level: Minimum - no cert data written to storage
```

The following example shows that restoration is from a PEM archive and that the database URL is flash:

```
Device# copy tftp://192.0.2.71/backup.ser flash:mycs.ser

Destination filename [mycs.ser]?
32 bytes copied in 1.320 secs (24 bytes/sec)
Router# copy tftp://192.0.2.71/backup.crl flash:mycs.crl
Destination filename [mycs.crl]?
214 bytes copied in 1.324 secs (162 bytes/sec)
Device# configure terminal

! Because CA cert has Digital Signature usage, you need to import using the "usage-keys"
keyword

Device(config)# crypto ca import mycs pem usage-keys terminal cisco123
% Enter PEM-formatted CA certificate.
% End with a blank line or "quit" on a line by itself.
! Paste the CA cert from .pem archive.
-----BEGIN CERTIFICATE-----
MIIB9zCCAWCgAwIBAgIBATANBgkqhkiG9w0BAQQFADAPMQ0wCwYDVQQDEwRteWNz
MB4XDTA0MDkwMjIxMDI1NloXDTA3MDkwMjIxMDI1NlowDzENMAsGA1UEAxMEbXlj
czCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAuGnnDXJbpDDQwCuKGs5Zg2rc
K7ZJauSUotTmWYQvNx+ZmWrUs5/j9Ee5FV2YonirGBQ9mc6u163kNlrIPFck062L
GpahBhNmKDgod1o2PHTnRlZpEZNDIqU2D3hACgByxPjrY4vUnccV36ewLnQnYpp8
szEu7PYTJr5dU5ltAekCAwEAAaNjMGEwDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8B
Af8EBAMCAYYwHwYDVR0jBBgwFoAUaEEQwYKCQ1dm9+wLYBKRTlzxaDIwHQYDVR0O
BBYEFGhBEMGCgkNXZvfsC2ASkU5c8WgyMA0GCSqGSIb3DQEBBAUAA4GBAHyhiv2C
mH+vswkBjRA1Fzzk8ttu9s5kwqG0dXp25QRUWsGlr9nsKPNdVKt3P7p0A/KochHe
eNiygiv+hDQ3FVnzsNv983le6O5jvAPxc17RO1BbfNhqvEWMsXdnjHOcUy7XerCo
+bdPcUf/eCiZueH/BEy/SZhD7yovzn2cdzBN
-----END CERTIFICATE-----
% Enter PEM-formatted encrypted private SIGNATURE key.
% End with "quit" on a line by itself.
! Paste the CA private key from .pem archive.
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,5053DC842B04612A

1CnlF5Pqvd0zp2NLZ7iosxzTy6nDeXPpNyJpxB5q+V29IuY8Apb6TlJCU7YrsEB/
nBTK7K76DCeGPlLpcuyEI171QmkQJ2gA0QhC0LrRo09WrINVH+b4So/y7nffZkVb
p2yDpZwqoJ8cmRH94Tie0YmzBtEh6ayOud11z53qbrsCnfSEwszt1xrW1MKrFZrk
/fTy6loHzGFzl3BDj4r5gBecExwcPp74ldHO+Ld4Nc9egG8BYkeBCsZZOQNVhXLN
I0tODOs6hP915zb6OrZFYv0NK6grTBO9D8hjNZ3U79jJzsSP7UNzIYHNTzRJiAyu
i56Oy/iHvkCSNUIK6zeIJQnW4bSoM1BqrbVPwHU6QaXUqlNzZ8SDtw7ZRZ/rHuiD
RTJMPbKquAzeuBss1132OaAUJRStjPXgyZTUbc+cWb6zATNws2yijPDTR6sRHoQL
47wHMr2Yj80VZGgkCSLAkL88ACz9TfUiVFhtfl6xMC2yuFl+WRk1XfF5VtWe5Zer
3Fn1DcBmlF7O86XUkiSHP4EV0cI6n5ZMzVLx0XAUtdAl1gD94y1V+6p9PcQHLyQA
pGRmj5IlSFw90aLafgCTbRbmC0ChIqHy91UFa1ub0130+yu7LsLGRlPmJ9NE61JR
bjRhlUXItRYWY7C4M3m/0wz6fmVQNSumJM08RHq6lUB3olzIgGIZlZkoaESrLG0p
qq2AENFemCPF0uhyVS2humMHjWuRr+jedfc/IMl7sLEgAdqCVCfV3RZVEaNXBud1
4QjkuTrwaTcRXVFbtrVioT/puyVUlpA7+k7w+F5TZwUV08mwvUEqDw==
-----END RSA PRIVATE KEY-----
quit
% Enter PEM-formatted SIGNATURE certificate.
% End with a blank line or "quit" on a line by itself.
! Paste the CA cert from .pem archive again.
-----BEGIN CERTIFICATE-----
MIIB9zCCAWCgAwIBAgIBATANBgkqhkiG9w0BAQQFADAPMQ0wCwYDVQQDEwRteWNz
MB4XDTA0MDkwMjIxMDI1NloXDTA3MDkwMjIxMDI1NlowDzENMAsGA1UEAxMEbXlj
czCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAuGnnDXJbpDDQwCuKGs5Zg2rc
K7ZJauSUotTmWYQvNx+ZmWrUs5/j9Ee5FV2YonirGBQ9mc6u163kNlrIPFck062L
GpahBhNmKDgod1o2PHTnRlZpEZNDIqU2D3hACgByxPjrY4vUnccV36ewLnQnYpp8
szEu7PYTJr5dU5ltAekCAwEAAaNjMGEwDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8B
Af8EBAMCAYYwHwYDVR0jBBgwFoAUaEEQwYKCQ1dm9+wLYBKRTlzxaDIwHQYDVR0O
BBYEFGhBEMGCgkNXZvfsC2ASkU5c8WgyMA0GCSqGSIb3DQEBBAUAA4GBAHyhiv2C
mH+vswkBjRA1Fzzk8ttu9s5kwqG0dXp25QRUWsGlr9nsKPNdVKt3P7p0A/KochHe
eNiygiv+hDQ3FVnzsNv983le6O5jvAPxc17RO1BbfNhqvEWMsXdnjHOcUy7XerCo
+bdPcUf/eCiZueH/BEy/SZhD7yovzn2cdzBN
-----END CERTIFICATE-----

% Enter PEM-formatted encrypted private ENCRYPTION key.
% End with "quit" on a line by itself.
! Because the CA cert only has Digital Signature usage, skip the encryption part.
quit
% PEM files import succeeded.
Device(config)# crypto pki server mycs
Device(cs-server)# database url flash:

! Fill in any certificate server configuration here.
Device(cs-server)# no shutdown

% Certificate Server enabled.
Device(cs-server)# end
Device# show crypto pki server

Certificate Server mycs:
    Status: enabled
    Server's current state: enabled
    Issuer name: CN=mycs
    CA cert fingerprint: F04C2B75 E0243FBC 19806219 B1D77412
    Granting mode is: manual
    Last certificate issued serial number: 0x2
    CA certificate expiration timer: 21:02:55 GMT Sep 2 2007
    CRL NextUpdate timer: 21:02:58 GMT Sep 9 2004
    Current storage dir: flash:
    Database Level: Minimum - no cert data written to storage
```

Example: Subordinate Certificate Server

The following configuration and output is typical of what you might see after configuring a subordinate certificate server. Please be aware that “ms2” refers to a 2048-bit RSA key that was generated in an earlier step.

```
Device(config)# crypto pki trustpoint sub
Device(ca-trustpoint)# enrollment url http://192.0.2.6
Device(ca-trustpoint)# rsa keypair ms2 2048
Device(ca-trustpoint)# exit
Device(config)# crypto pki server sub
Device(cs-server)# mode sub-cs
Device(ca-server)# no shutdown

%Some server settings cannot be changed after CA certificate generation.
```

```
% Please enter a passphrase to protect the private key
% or type Return to exit
Password:
Jan  6 22:32:22.698: CRYPTO_CS: enter FSM: input state initial, input signal no shut
Re-enter password:
Jan  6 22:32:30.302: CRYPTO_CS: starting enabling checks
Jan  6 22:32:30.306: CRYPTO_CS: key 'sub' does not exist; generated automatically [OK]
Jan  6 22:32:39.810: %SSH-5-ENABLED: SSH 1.99 has been enabled
Certificate has the following attributes:
     Fingerprint MD5: 328ACC02 52B25DB8 22F8F104 B6055B5B
     Fingerprint SHA1: 02FD799D DD40C7A8 61DC53AB 1E89A3EA 2A729EE2
% Do you accept this certificate? [yes/no]:
Jan  6 22:32:44.830: CRYPTO_CS: nvram filesystem
Jan  6 22:32:44.922: CRYPTO_CS: serial number 0x1 written.
Jan  6 22:32:46.798: CRYPTO_CS: created a new serial file.
Jan  6 22:32:46.798: CRYPTO_CS: authenticating the CA 'sub'y
Trustpoint CA certificate accepted.%
% Certificate request sent to Certificate Authority
% Enrollment in progress...
Router (cs-server)#
Jan  6 22:33:30.562: CRYPTO_CS: Publishing 213 bytes to crl file nvram:sub.crl
Jan  6 22:33:32.450: CRYPTO_CS: enrolling the server's trustpoint 'sub'
Jan  6 22:33:32.454: CRYPTO_CS: exit FSM: new state check failed
Jan  6 22:33:32.454: CRYPTO_CS: cs config has been locked
Jan  6 22:33:33.118: CRYPTO_PKI:  Certificate Request Fingerprint MD5: CED89E5F 53B9C60E
> AA123413 CDDAD964
Jan  6 22:33:33.118: CRYPTO_PKI:  Certificate Request Fingerprint SHA1: 70787C76 ACD7E67F
7D2C8B23 98CB10E7 718E84B1
% Exporting Certificate Server signing certificate and keys...
Jan  6 22:34:53.839: %PKI-6-CERTRET: Certificate received from Certificate Authority
Jan  6 22:34:53.843: CRYPTO_CS: enter FSM: input state check failed, input signal cert
configured
Jan  6 22:34:53.843: CRYPTO_CS: starting enabling checks
Jan  6 22:34:53.843: CRYPTO_CS: nvram filesystem
Jan  6 22:34:53.883: CRYPTO_CS: found existing serial file.
Jan  6 22:34:53.907: CRYPTO_CS: old router cert flag 0x4
Jan  6 22:34:53.907: CRYPTO_CS: new router cert flag 0x44
Jan  6 22:34:56.511: CRYPTO_CS: DB version
Jan  6 22:34:56.511: CRYPTO_CS: last issued serial number is 0x1
Jan  6 22:34:56.551: CRYPTO_CS: CRL file sub.crl exists.
Jan  6 22:34:56.551: CRYPTO_CS: Read 213 bytes from crl file sub.crl.
Jan  6 22:34:56.603: CRYPTO_CS: SCEP server started
Jan  6 22:34:56.603: CRYPTO_CS: exit FSM: new state enabled
Jan  6 22:34:56.603: CRYPTO_CS: cs config has been locked
Jan  6 22:35:02.359: CRYPTO_CS: enter FSM: input state enabled, input signal time set
Jan  6 22:35:02.359: CRYPTO_CS: exit FSM: new state enabled
Jan  6 22:35:02.359: CRYPTO_CS: cs config has been locked
```

Example: Root Certificate Server Differentiation

When issuing certificates, the root certificate server (or parent subordinate certificate server) differentiates the certificate request from “Sub CA,” “RA,” and peer requests, as shown in the following sample output:

```
Device# crypto pki server server1 info req

Enrollment Request Database:
RA certificate requests:
ReqID     State        Fingerprint                               SubjectName
----------------------------------------------------------------------------
Subordinate CS certificate requests:
ReqID     State        Fingerprint                               SubjectName
----------------------------------------------------------------------------
1     pending        CB9977AD8A73B146D3221749999B0F66 hostname=host-subcs.company.com
RA certificate requests:
```

```
ReqID     State        Fingerprint                                SubjectName
-----------------------------------------------------------------------------
Router certificate requests:
ReqID     State        Fingerprint                                SubjectName
-----------------------------------------------------------------------------
```

Example: Show Output for a Subordinate Certificate Server

The following **show crypto pki server command**output indicates that a subordinate certificate server has been configured:

```
Device# show crypto pki server

Certificate Server sub:
  Status: enabled
  Server's configuration is locked  (enter "shut" to unlock it)
  Issuer name: CN=sub
  CA cert fingerprint: 11B586EE 3B354F33 14A25DDD 7BD39187
  Server configured in subordinate server mode
  Upper CA cert fingerprint: 328ACC02 52B25DB8 22F8F104 B6055B5B
  Granting mode is: manual
  Last certificate issued serial number: 0x1
  CA certificate expiration timer: 22:33:44 GMT Jan 6 2006
  CRL NextUpdate timer: 22:33:29 GMT Jan 13 2005
  Current storage dir: nvram:
  Database Level: Minimum - no cert data written to storage
```

Example: RA Mode Certificate Server

The following output is typical of what you might see after having configured an RA mode certificate server:

```
Device-ra(config)# crypto pki trustpoint myra
Device-ra(ca-trustpoint)# enrollment url http://192.0.2.17
! Include "cn=ioscs RA" or "ou=ioscs RA" in the subject-name.
Device-ra(ca-trustpoint)# subject-name cn=myra, ou=ioscs RA, o=company, c=us
Device-ra(ca-trustpoint)# exit
Device-ra(config)# crypto pki server myra
Device-ra(cs-server)# mode ra
Device-ra(cs-server)# no shutdown

% Generating 1024 bit RSA keys ...[OK]
Certificate has the following attributes:
Fingerprint MD5: 32661452 0DDA3CE5 8723B469 09AB9E85
Fingerprint SHA1: 9785BBCD 6C67D27C C950E8D0 718C7A14 C0FE9C38
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Ready to request the CA certificate.
%Some server settings cannot be changed after the CA certificate has been requested.
Are you sure you want to do this? [yes/no]: yes
%
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this
   password to the CA administrator in order to revoke your certificate.
   For security reasons your password will not be saved in the configuration.
   Please make a note of it.
Password:
Re-enter password:
% The subject name in the certificate will include: cn=myra, ou=ioscs RA, o=company, c=us
% The subject name in the certificate will include: Router-ra.company.com
% Include the router serial number in the subject name? [yes/no]: no
% Include an IP address in the subject name? [no]: no
Request certificate from CA? [yes/no]: yes
% Certificate request sent to Certificate Authority
% The certificate request fingerprint will be displayed.
% The 'show crypto pki certificate' command will also show the fingerprint.
% Enrollment in progress...
Device-ra (cs-server)#

Sep 15 22:32:40.197: CRYPTO_PKI:  Certificate Request Fingerprint MD5: 82B41A76 AF4EC87D
AAF093CD 07747D3A
Sep 15 22:32:40.201: CRYPTO_PKI:  Certificate Request Fingerprint SHA1: 897CDF40 C6563EAA
0FED05F7 0115FD3A 4FFC5231
Sep 15 22:34:00.366: %PKI-6-CERTRET: Certificate received from Certificate Authority

Device-ra(cs-server)# end
Device-ra# show crypto pki server

Certificate Server myra:
    Status: enabled
    Issuer name: CN=myra
    CA cert fingerprint: 32661452 0DDA3CE5 8723B469 09AB9E85
    ! Note that the certificate server is running in RA mode
    Server configured in RA mode
    RA cert fingerprint: C65F5724 0E63B3CC BE7AE016 BE0D34FE
    Granting mode is: manual
    Current storage dir: nvram:
    Database Level: Minimum - no cert data written to storage
```

The following output shows the enrollment request database of the issuing certificate server after the RA has been enabled:

Note The RA certificate request is recognized by the issuing certificate server because "ou=ioscs RA" is listed in the subject name.

```
Device-ca# crypto pki server mycs info request

Enrollment Request Database:
Subordinate CA certificate requests:
ReqID  State      Fingerprint                      SubjectName
--------------------------------------------------------------
! The request is identified as RA certificate request.
RA certificate requests:
ReqID  State      Fingerprint                      SubjectName
--------------------------------------------------------------
12     pending    88F547A407FA0C90F97CDE8900A30CB0
hostname=Router-ra.company.com,cn=myra,ou=ioscs RA,o=company,c=us
Router certificates requests:
ReqID   State     Fingerprint                      SubjectName
--------------------------------------------------------------
! Issue the RA certificate.
Device-ca# crypto pki server mycs grant 12
```

The following output shows that the issuing certificate server is configured to issue a certificate automatically if the request comes from an RA:

```
Device-ca(config)# crypto pki server mycs
Device-ca(cs-server)# grant ra-auto

% This will cause all certificate requests already authorized by known RAs to be automatically
```

```
 granted.
Are you sure you want to do this? [yes/no]: yes
Router-ca (cs-server)# end
Device-ca# show crypto pki server

Certificate Server mycs:
    Status: enabled
    Server's current state: enabled
    Issuer name: CN=mycs
    CA cert fingerprint: 32661452 0DDA3CE5 8723B469 09AB9E85
    ! Note that the certificate server will issue certificate for requests from the RA.
    Granting mode is: auto for RA-authorized requests, manual otherwise
    Last certificate issued serial number: 0x2
    CA certificate expiration timer: 22:29:37 GMT Sep 15 2007
    CRL NextUpdate timer: 22:29:39 GMT Sep 22 2004
    Current storage dir: nvram:
    Database Level: Minimum - no cert data written to storage
```

The following example shows the configuration of “myra”, an RA server, configured to support automatic rollover from “myca”, the CA. After the RA server is configured, automatic granting of certificate reenrollment requests is enabled:

```
crypto pki trustpoint myra
 enrollment url
http://myca
 subject-name ou=iosca RA
 rsakeypair myra
crypto pki server myra
 mode ra
 auto-rollover
crypto pki server mycs
 grant auto rollover ra-cert
 auto-rollover 25
```

Example: Enabling CA Certificate Rollover to Start Immediately

The following example shows how to enable automated CA certificate rollover on the server mycs with the **crypto pki server**command. The **show crypto pki server** command then shows the current state of the mycs server and that the rollover certificate is currently available for rollover.

```
Device(config)# crypto pki server mycs rollover

Jun 20 23:51:21.211:%PKI-4-NOSHADOWAUTOSAVE:Configuration was
modified.  Issue "write memory" to save new IOS CA certificate
! The config has not been automatically saved because the config has been changed.
Device# show crypto pki server

Certificate Server mycs:
    Status:enabled
    Server's configuration is locked  (enter "shut" to unlock it)
    Issuer name:CN=mycs
    CA cert fingerprint:E7A5FABA 5D7AA26C F2A9F7B3 03CE229A
    Granting mode is:manual
    Last certificate issued serial number:0x2
    CA certificate expiration timer:00:49:26 PDT Jun 20 2008
    CRL NextUpdate timer:00:49:29 PDT Jun 28 2005
    Current storage dir:nvram:
    Database Level:Minimum - no cert data written to storage
    Rollover status:available for rollover
    ! Rollover certificate is available for rollover.
    Rollover CA certificate fingerprint:9BD7A443 00A6DD74 E4D9ED5F B7931BE0
    Rollover CA certificate expiration time:00:49:26 PDT Jun 20 2011
    Auto-Rollover configured, overlap period 25 days
```

Where to Go Next

After the certificate server is successfully running, you can either begin enrolling clients through manual mechanisms (as explained in the module "*Configuring Certificate Enrollment for a PKI*") or begin configuring SDP, which is a web-based enrollment interface, (as explained in the module "*Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI*.")

Additional References for Configuring and Managing a Certificate Server for PKI Deployment

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
PKI and security commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z
USB Token RSA Operations: Using the RSA keys on a USB token for initial autoenrollment	*Configuring Certificate Enrollment for a PKI*
USB Token RSA Operations: Benefits of using USB tokens	*Storing PKI Credentials*
Certificate server client certificate enrollment, autoenrollment, and automatic rollover	*Configuring Certificate Enrollment for a PKI*
Setting up and logging into a USB token	*Storing PKI Credentials*
Web-based certificate enrollment	*Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI*
RSA keys in PEM formatted files	*Deploying RSA Keys Within a PKI*
Choosing a certificate revocation mechanism	*Configuring Authorization and Revocation of Certificates in a PKI*

Related Topic	Document Title
Recommended cryptographic algorithms	Next Generation Encryption

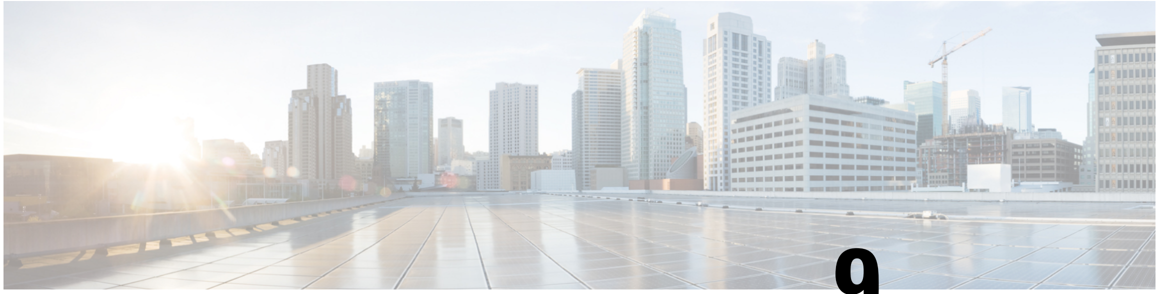
Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring and Managing a Certificate Server for PKI Deployment

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.



CHAPTER 9

Storing PKI Credentials

Public key infrastructure (PKI) credentials, such as Rivest, Shamir, and Adelman (RSA) keys and certificates can be stored in a specific location on the router, such as NVRAM and flash memory or on a USB eTtoken 64 KB smart card. USB tokens provide secure configuration distribution, RSA operations such as on-token key generation, signing, and authentication, and the storage of Virtual Private Network (VPN) credentials for deployment.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Storing PKI Credentials

Prerequisites for Specifying a Local Certificate Storage Location

Before you can specify the local certificate storage location, your system should meet the following requirements:

- A Cisco IOS Release 12.4(2)T PKI-enabled image or a later image
- A platform that supports storing PKI credentials as separate files

- A configuration that contains at least one certificate
- An accessible local file system

Prerequisites for Specifying USB Token Storage for PKI Credentials

Before you can use a USB token, your system should meet the following requirements:

- A Cisco 871 router, Cisco 1800 series, Cisco 2800 series, a Cisco 3800 series router, or a Cisco 7200VXR NPE-G2 platform
- At least a Cisco IOS Release 12.3(14)T image running on any of the supported platforms
- A Cisco supported USB token (Safenet/Aladdin eToken PRO 32 KB or 64 KB)
- A k9 image

Restrictions for Storing PKI Credentials

Restrictions for Specifying a Local Certificate Storage Location

When storing certificates to a local storage location, the following restrictions are applicable:

- Only local file systems may be used. An error message will be displayed if a remote file system is selected, and the command will not take effect.
- A subdirectory may be specified if supported by the local file system. NVRAM does not support subdirectories.

Restrictions for Specifying USB Token Storage

When using a USB token to store PKI data, the following restrictions are applicable:

- USB token support requires a 3DES (k9) Cisco IOS software image, which provides secure file storage.
- You cannot boot an image from a USB token. (However, you can boot a configuration from a USB token.)
- USB hubs are currently not supported. Thus, the number of supported devices is limited to the number of available USB ports.

Information About Storing PKI Credentials

Storing Certificates to a Local Storage Location

Certificates are stored to NVRAM by default; however, some routers do not have the required amount of NVRAM to successfully store certificates.

All Cisco platforms support NVRAM and flash local storage. Depending on your platform, you may have other supported local storage options including bootflash, slot, disk, USB flash, or USB token.

During run time, you can specify what active local storage device you would like to use to store certificates.

PKI Credentials and USB Tokens

To use a secure USB token on your router, you should understand the following concepts:

How a USB Token Works

A smart card is a small plastic card, containing a microprocessor and memory that allows you to store and process data. A USB token is a smart card with a USB interface. The token can securely store any type of file within its available storage space (32 KB). Configuration files that are stored on the USB token can be encrypted and accessed only via a user PIN. The device does not load the configuration file unless the proper PIN has been configured for secure deployment of device configuration files.

After you plug the USB token into the device, you must log into the USB token; thereafter, you can change default settings, such as the user PIN (default: 1234567890) and the allowed number of failed login attempts (default: 15 attempts) before future logins are refused. For more information on accessing and configuring the USB token, see the section “Logging Into and Setting Up the USB Token."

After you have successfully logged into the USB token, you can copy files from the device on to the USB token via the **copy** command. USB token RSA keys and associated IPsec tunnels remain available until the device is reloaded. To specify the length of time before the keys are removed and the IPsec tunnels are torn down, issue the **crypto pki token removal timeout** command. The default timeout is zero, which causes the RSA keys to be removed automatically after the eToken is removed from the device. The default appears in the running configuration as:

```
crypto pki token default removal timeout 0
```

The table below highlights the capabilities of the USB token.

Table 12: Functionality Highlights for USB Tokens

Function	USB Token
Accessibility	Used to securely store and transfer digital certificates, preshared keys, and device configurations from the USB token to the device.
Storage Size	32 KB or 64 KB
File Types	• Typically used to store digital certificates, preshared keys, and device configurations for IPsec VPNs. • USB tokens cannot store Cisco IOS images.
Security	• Files can be encrypted and accessed only with a user PIN. • Files can also be stored in a nonsecure format.
Boot Configurations	• The device can use the configuration stored in the USB token during boot time. • The device can use the secondary configuration stored in the USB token during boot time. (A secondary configuration allows users to load their IPsec configuration.)

Benefits of USB Tokens

USB token support on a Cisco router provides the following application benefits:

Removable Credentials: Provide or Store VPN Credentials on an External Device for Deployment

A USB token can use smart card technology to store a digital certificate and configuration for IPsec VPN deployment. This ability enhances the capability of the router to generate RSA public keys to authenticate at least one IPsec tunnel. (Because a router can initiate multiple IPsec tunnels, the USB token can contain several certificates, as appropriate.)

Storing VPN credentials on an external device reduces the threat of compromising secure data.

PIN Configuration for Secure File Deployment

A USB token can store a configuration file that can be used for enabling encryption on the router via a user-configured PIN. (That is, no digital certificates, preshared keys, or VPNs are used.)

Touchless or Low Touch Configuration

The USB token can provide remote software configuration and provisioning with little or no human interaction. Configuration is set up as an automated process. That is, the USB token can store a bootstrap configuration that the router can use to boot from after the USB token has been inserted into the router. The bootstrap configuration connects the router to a TFTP server, which contains a configuration that completely configures the router.

RSA Operations

A USB token may be used as a cryptographic device in addition to a storage device. Using a USB token as a cryptographic device allows RSA operations such as key generation, signing, and authentication to be performed on the token.

General-purpose, special-usage, encryption, or signature RSA key pairs with a modulus of 2048 bits or less may be generated from credentials located on your token storage device. Private keys are not distributed and remain on the token by default, however you may configure the private key storage location.

Keys that reside on a USB token are saved to persistent token storage when they are generated. Key deletion will remove the keys stored on the token from persistent storage immediately. (Keys that do not reside on a token are saved to or deleted from non-token storage locations when the **write memory** or a similar command is issued.)

Remote Device Configuration and Provisioning in a Secure Device Provisioning (SDP) Environment

SDP may be used to configure a USB token. The configured USB token may be transported to provision a device at a remote location. That is, a USB token may be used to transfer cryptographic information from one network device to another remote network device providing a solution for a staged USB token deployment.

For information about using USB tokens with SDP, see document titles in the “Additional References” section.

How to Configure PKI Storage

Specifying a Local Storage Location for Certificates

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki certificate storage** *location-name*
4. **exit**
5. **copy** *source-url destination-url*
6. **show crypto pki certificates storage**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki certificate storage** *location-name* **Example:** `Device(config)# crypto pki certificate storage flash:/certs`	Specifies the local storage location for certificates.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

Step 5	**copy** *source-url destination-url*
	Example:
	`Device#`
	`copy system:running-config nvram:startup-config`
		(Optional) Saves the running configuration to the startup configuration.
		Note Settings will only take effect when the running configuration is saved to the startup configuration.

Step 6	**show crypto pki certificates storage** **Example:**	(Optional) Displays the current setting for the PKI certificate storage location.

	Command or Action	Purpose
	`Device# show crypto pki certificates storage`	

Example

The following is sample output from the **show crypto pki certificates storage** command, which shows that the certificates are stored in the certs subdirectory of disk0:

```
Device# show crypto pki certificates storage
Certificates will be stored in disk0:/certs/
```

Setting Up and Using USB Tokens on Cisco Devices

Storing the Configuration on a USB Token

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **boot config** *usbtoken[0-9]:filename*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**boot config** *usbtoken[0-9]:filename* **Example:** `Device(config)# boot config usbtoken0:file`	Specifies that the startup configuration file is stored in a secure USB token.

Logging Into and Setting Up the USB Token

How RSA Keys are Used with a USB Token

- RSA keys are loaded after the USB token is successfully logged into the router.

- By default, newly generated RSA keys are stored on the most recently inserted USB token. Regenerated keys should be stored in the same location where the original RSA key was generated.

Configuring the Device for Manual Login

Unlike automatic login, manual login requires that the user know the actual USB token PIN.

Note Either the manual or automatic login is required.

Manual login can be used when storing a PIN on the device is not desirable. Manual login may also be suitable for some initial deployment or hardware replacement scenarios for which the device is obtained from the local supplier or drop-shipped to the remote site. Manual login can be executed with or without privileges, and it creates files and RSA keys on the USB token available to the Cisco IOS software. If a secondary configuration file is configured, it is executed only with the privileges of the user who is performing the login. Thus, if you want to use manual login and set up the secondary configuration on the USB token to perform anything useful, you need to enable privileges.

Manual login can also be used in recovery scenarios for which the device configuration has been lost. If the scenario contains a remote site that normally connects to the core network with a VPN, the loss of the configuration and RSA keys requires out-of-band services that the USB token can provide. The USB token can contain a boot configuration, a secondary configuration, or both, and RSA keys to authenticate the connection.

SUMMARY STEPS

1. **enable**
2. **crypto pki token** *token-name* [**admin**] **login** [*pin*]
3. **show usbtoken** *0-9***:***filename*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**crypto pki token** *token-name* [**admin**] **login** [*pin*] **Example:** `Device# crypto pki token usbtoken0 admin login 5678`	Manually logs into the USB token. If the **admin** keyword is not specified initially you can re-enter the **crypto pki token** command again with this keyword option.
Step 3	**show usbtoken** *0-9***:***filename* **Example:** `Device# show usbtoken0:usbfile`	(Optional) Verifies whether the USB token has been logged on to the device.

What to Do Next

After you have logged into the USB token, it is available for use.

- To further configure the USB token, see the “Configuring the USB Token” section.
- To perform USB token administrative tasks, such as changing the user PIN, copying files from the router to the USB token set key storage location, and changing USB tokens, see the “Setting Administrative Functions on the USB Token” section.

Configuring the USB Token

After you have set up automatic login, you may perform this task to further configure the USB token.

PINs and Passphrases

For additional PIN security with automatic login, you may encrypt your PIN stored in NVRAM and set up a passphrase for your USB token. Establishing a passphrase allows you to keep your PIN secure; another user needs only to know the passphrase, not the PIN.

When the USB token is inserted into the device, the passphrase is needed to decrypt the PIN. Once the PIN is decrypted, the device can then use the PIN to log in to the USB token.

Note The user needs a privilege level of 1 to log in.

Unlocking and Locking the USB Token

The USB token itself can be locked (encrypted) or unlocked (decrypted).

Unlocking the USB token allows it to be used. Once unlocked, Cisco IOS software treats the token as if it were automatically logged in. Any keys on the USB token are loaded, and if a secondary configuration file is on the token, it is executed with full user privileges (privilege level 15) independent of the privilege level of the logged-in user.

Locking the token, unlike logging out of the token, deletes any RSA keys loaded from the token and runs the secondary unconfiguration file, if configured.

Secondary Configuration and Unconfiguration Files

Configuration files that exist on a USB token are called secondary configuration files. If you create and configure a secondary configuration file, it is executed after the token is logged in. The existence of a secondary configuration file is determined by the presence of a secondary configuration file option in the Cisco IOS configuration stored in NVRAM. When the token is removed or logged out and the removal timer expires, a separate secondary unconfiguration file is processed to remove all secondary configuration elements from the running configuration. Secondary configuration and secondary unconfiguration files are executed at privilege level 15 and are not dependent on the level of the user logged in.

SUMMARY STEPS

1. **enable**
2. **crypto pki token** *token-name* **unlock** [*pin*]
3. **configure terminal**
4. **crypto pki token** *token-name* **encrypted-user-pin** [**write**]

5. **crypto pki token** *token-name* **secondary unconfig** *file*
6. **exit**
7. **crypto pki token** *token-name* **lock** [*pin*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**crypto pki token** *token-name* **unlock** [*pin*] **Example:** `Device# crypto pki token mytoken unlock mypin`	(Optional) Allows the token to be used if the USB token has been locked. Once unlocked, Cisco IOS software treats the token as if it has been automatically logged in. Any keys on the token are loaded and if a secondary configuration file exists, it is executed.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**crypto pki token** *token-name* **encrypted-user-pin** [**write**] **Example:** `Device(config)# crypto pki token mytoken encrypted-user-pin write`	(Optional) Encrypts the stored PIN in NVRAM.
Step 5	**crypto pki token** *token-name* **secondary unconfig** *file* **Example:** `Device(config)# crypto pki token mytoken secondary unconfig configs/myunconfigfile.cfg`	(Optional) Specifies the secondary configuration file and its location.
Step 6	**exit** **Example:** `Device(config)# exit`	Enters privileged EXEC mode.
Step 7	**crypto pki token** *token-name* **lock** [*pin*] **Example:** `Device# crypto pki token mytoken lock mypin`	(Optional) Deletes any RSA keys loaded from the token and runs the secondary unconfiguration file, if it exists.

Examples

The following example shows both the configuration and encryption of a user PIN and then the device reloading and the user PIN being unlocked:

```
! Configuring the user PIN

Enter configuration commands, one per line. End with CNTL/Z.

Device(config)# crypto pki token usbtoken0: userpin

Enter password: mypassword

! Encrypt the user PIN

Device(config)# crypto pki token usbtoken0: encrypted-user-pin

Enter passphrase: mypassphrase

Device(config)# exit

Device#

Sep 20 21:51:38.076: %SYS-5-CONFIG_I: Configured from console by console

Device# show running config

crypto pki token usbtoken0 user-pin *encrypted*

! Reloading the router.

Device> enable

Password:

! Decrypting the user pin.

Device# crypto pki token usbtoken0: unlock

Token eToken is usbtoken0

Enter passphrase: mypassphrase

Token login to usbtoken0(eToken) successful

Device#

Sep 20 22:31:13.128: %CRYPTO-6-TOKENLOGIN: Cryptographic Token eToken

Login Successful
```

The following example shows a how a secondary unconfiguration file might be used to remove secondary configuration elements from the running configuration. For example, a secondary configuration file might be used to set up a PKI trustpoint. A corresponding unconfiguration file, named mysecondaryunconfigfile.cfg, might contain this command line:

```
no crypto pki trustpoint token-tp
```

If the token were removed and the following commands executed, the trustpoint and associated certificates would be removed from the device’s running configuration:

```
 Device# configure terminal
Device(config)# no crypto pki token mytoken secondary unconfig mysecondaryunconfigfile.cfg
```

What to Do Next

After you have logged into and configured the USB token, it is available for use. If you want to perform USB token administrative tasks, such as changing the user PIN, copying files from the router to the USB token set key storage location, and changing USB tokens, see the “Setting Administrative Functions on the USB Token” section.

Setting Administrative Functions on the USB Token

Perform this task to change default settings, such as the user PIN, the maximum number of failed attempts on the USB token, or the credential storage location.

SUMMARY STEPS

1. **enable**
2. **crypto pki token** *token-name* **admin**] **change-pin** [*pin*]
3. **crypto pki token** *token-name device-name***: label** *token-label*
4. **configure terminal**
5. **crypto key storage** *device-name***:**
6. **crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *device-name***:**] [**redundancy**] [**on** *device-name*]**:**
7. **crypto key move rsa** *keylabel* [**non-exportable** | [**on** | **storage**]] *location*
8. **crypto pki token** {*token-name* | **default**} **removal timeout** [*seconds*]
9. **crypto pki token** {*token-name* | **default**} **max-retries** [*number*]
10. **exit**
11. **copy usbflash**[*0-9*]**:***filename destination-url*
12. **show usbtoken**[*0-9*]**:***filename*
13. **crypto pki token** *token-name* **logout**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**crypto pki token** *token-name* **admin**] **change-pin** [*pin*] **Example:** `Device# crypto pki token usbtoken0 admin change-pin`	(Optional) Changes the user PIN number on the USB token. • If the PIN is not changed, the default PIN 1234567890 is used.

	Command or Action	Purpose
		Note After the PIN has been changed, you must reset the login failure count to zero (via the **crypto pki token max-retries** command). The maximum number of allowable login failures is set (by default) to 15.
Step 3	**crypto pki token** *token-name device-name***: label** *token-label* **Example:** `Device# crypto pki token mytoken usb0: label newlabel`	(Optional) Sets or changes the name of the USB token. - The value of the *token-label* argument may be up to 31 alphanumeric characters in length including dashes and underscores. **Tip** This command is useful when configuring multiple USB tokens for automatic login, secondary configuration files, or other token specific settings.
Step 4	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 5	**crypto key storage** *device-name***:** **Example:** `Device(config)# crypto key storage usbtoken0:`	(Optional) Sets the default RSA key storage location for newly created keys. **Note** Regardless of configuration settings, existing keys are stored on the device from where they were originally loaded.
Step 6	**crypto key generate rsa** [**general-keys** | **usage-keys** | **signature** | **encryption**] [**label** *key-label*] [**exportable**] [**modulus** *modulus-size*] [**storage** *device-name***:**] [**redundancy**] [**on** *device-name*]**:** **Example:** `Device(config)# crypto key generate rsa label tokenkey1 storage usbtoken0:`	(Optional) Generates the RSA key pair for the certificate server. - The **storage** keyword specifies the key storage location. - When specifying a label name by specifying the *key-label* argument, you must use the same name for the label that you plan to use for the certificate server (through the **crypto pki server** *cs-label* command). If a *key-label* argument is not specified, the default value, which is the fully qualified domain name (FQDN) of the device, is used. If the exportable RSA key pair is manually generated after the CA certificate has been generated, and before issuing the **no shutdown** command, then use the **crypto ca export pkcs12** command to export a PKCS12 file that contains the certificate server certificate and the private key. - By default, the modulus size of a CA key is 1024 bits. The recommended modulus for a CA key is 2048 bits.

	Command or Action	Purpose
		The range for a modulus size of a CA key is from 350 to 4096 bits. • The **on** keyword specifies that the RSA key pair is created on the specified device, including a Universal Serial Bus (USB) token, local disk, or NVRAM. The name of the device is followed by a colon (:). **Note** Keys created on a USB token must be 2048 bits or less.
Step 7	**crypto key move rsa** *keylabel* [**non-exportable** | [**on** | **storage**]] *location* **Example:** `Device(config)# crypto key move rsa keypairname non-exportable on token`	(Optional) Moves existing Cisco IOS credentials from the current storage location to the specified storage location. By default, the RSA key pair remains stored on the current device. Generating the key on the device and moving it to the token takes less than a minute. Generating a key on the token, using the **on** keyword could take five to ten minutes, and is dependent on hardware key generation routines available on the USB token. When an existing RSA key pair is generated in Cisco IOS, stored on a USB token, and used for an enrollment, it may be necessary to move those existing RSA key pairs to an alternate location for permanent storage. This command is useful when using SDP with USB tokens to deploy credentials.
Step 8	**crypto pki token** {*token-name* | **default**} **removal timeout** [*seconds*] **Example:** `Device(config)# crypto pki token usbtoken0 removal timeout 60`	(Optional) Sets the time interval, in seconds, that the device waits before removing the RSA keys that are stored in the USB token after the USB token has been removed from the device. **Note** If this command is not issued, all RSA keys and IPsec tunnels associated with the USB token are torn down immediately after the USB token is removed from the device.
Step 9	**crypto pki token** {*token-name* | **default**} **max-retries** [*number*] **Example:** `Device(config)# crypto pki token usbtoken0 max-retries 20`	(Optional) Sets the maximum number of consecutive failed login attempts allowed before access to the USB token is denied. • By default, the value is set at 15.
Step 10	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

	Command or Action	Purpose
Step 11	**copy usbflash**[*0-9*]**:***filename destination-url* **Example:** `Device# copy usbflash0:file1 nvram:`	Copies files from USB token to the device. • *destination-url*—See the **copy** command page documentation for a list of supported options.
Step 12	**show usbtoken**[*0-9*]**:***filename* **Example:** `Device# show usbtoken:usbfile`	(Optional) Displays information about the USB token. You can use this command to verify whether the USB token has been logged in to the device.
Step 13	**crypto pki token** *token-name* **logout** **Example:** `Device# crypto pki token usbtoken0 logout`	Logs the device out of the USB token. **Note** If you want to save any data to the USB token, you must log back into the token.

Troubleshooting USB Tokens

This section contains descriptions of the following Cisco IOS commands that can be used to help troubleshoot possible problems that may arise while using a USB token:

Troubleshooting the USB Port Connection

Use the **show file systems** command to determine whether the router recognizes that there is a USB module plugged into a USB port. The USB module should appear on the list of file systems. If the module does not appear on the list, it can indicate any of the following problems:

- A connection problem with the USB module.
- The Cisco IOS image running on the router does not support a USB module.
- A hardware problem with the USB module itself.

Sample output from the **show file systems** command showing a USB token appears below. The USB module listing appears in the last line of the examples.

```
Device# show file systems
File Systems:
     Size(b)     Free(b)      Type  Flags  Prefixes
           -           -    opaque     rw   archive:
           -           -    opaque     rw   system:
           -           -    opaque     rw   null:
           -           -   network     rw   tftp:
*  129880064    69414912      disk     rw   flash:#
      491512      486395     nvram     rw   nvram:
           -           -    opaque     wo   syslog:
           -           -    opaque     rw   xmodem:
           -           -    opaque     rw   ymodem:
           -           -   network     rw   rcp:
           -           -   network     rw   pram:
           -           -   network     rw   ftp:
           -           -   network     rw   http:
           -           -   network     rw   scp:
```

```
         -           -   network     rw   https:
         -           -    opaque     ro   cns:
  63158272    33037312  usbflash     rw   usbflash0:
     32768         858  usbtoken     rw   usbtoken1:
```

Determining if a USB Token is Supported by Cisco

Use the **show usb device** command to determine if a USB token is supported by Cisco. The following output from this command indicates whether or not the module is supported is bold in the sample output below:

```
Router# show usb device
Host Controller:1
Address:0x11
Device Configured:YES
Device Supported:YES
Description:eToken Pro 4254
Manufacturer:AKS
Version:1.0
Serial Number:
Device Handle:0x1010000
USB Version Compliance:1.0
Class Code:0xFF
Subclass Code:0x0
Protocol:0x0
Vendor ID:0x529
Product ID:0x514
Max. Packet Size of Endpoint Zero:8
Number of Configurations:1
Speed:Low
Selected Configuration:1
Selected Interface:0
Configuration:
    Number:1
    Number of Interfaces:1
    Description:
    Attributes:None
    Max Power:60 mA
    Interface:
        Number:0
        Description:
        Class Code:255
        Subclass:0
        Protocol:0
        Number of Endpoints:0
```

Determining USB Token Device Problems

Use the **show usb controllers** command to determine if there is a hardware problem with a USB flash module. If the **show usb controllers** command displays an error, the error indicates a hardware problem in the USB module.

You can also use the **show usb controllers** command to verify that copy operations onto a USB flash module are occurring successfully. Issuing the **show usb controllers** command after performing a file copy should display successful data transfers.

The following sample output for the **show usb controllers** command displays a working USB flash module:

```
Router# show usb controllers
Name:1362HCD
Controller ID:1
Controller Specific Information:
    Revision:0x11
    Control:0x80
    Command Status:0x0
    Hardware Interrupt Status:0x24
    Hardware Interrupt Enable:0x80000040
    Hardware Interrupt Disable:0x80000040
    Frame Interval:0x27782EDF
    Frame Remaining:0x13C1
    Frame Number:0xDA4C
    LSThreshold:0x628
    RhDescriptorA:0x19000202
    RhDescriptorB:0x0
    RhStatus:0x0
    RhPort1Status:0x100103
    RhPort2Status:0x100303
    Hardware Configuration:0x3029
    DMA Configuration:0x0
    Transfer Counter:0x1
    Interrupt:0x9
    Interrupt Enable:0x196
    Chip ID:0x3630
    Buffer Status:0x0
    Direct Address Length:0x80A00
    ATL Buffer Size:0x600
    ATL Buffer Port:0x0
    ATL Block Size:0x100
    ATL PTD Skip Map:0xFFFFFFFF
    ATL PTD Last:0x20
    ATL Current Active PTD:0x0
    ATL Threshold Count:0x1
    ATL Threshold Timeout:0xFF
Int Level:1
Transfer Completion Codes:
         Success              :920             CRC             :0
         Bit Stuff            :0               Stall           :0
         No Response          :0               Overrun         :0
         Underrun             :0               Other           :0
         Buffer Overrun       :0               Buffer Underrun :0
Transfer Errors:
         Canceled Transfers   :2               Control Timeout :0
Transfer Failures:
         Interrupt Transfer   :0               Bulk Transfer   :0
         Isochronous Transfer :0               Control Transfer:0
Transfer Successes:
         Interrupt Transfer   :0               Bulk Transfer   :26
         Isochronous Transfer :0               Control Transfer:894
USBD Failures:
         Enumeration Failures :0               No Class Driver Found:0
         Power Budget Exceeded:0
USB MSCD SCSI Class Driver Counters:
         Good Status Failures :3               Command Fail    :0
         Good Status Timed out:0               Device not Found:0
         Device Never Opened  :0               Drive Init Fail :0
         Illegal App Handle   :0               Bad API Command :0
         Invalid Unit Number  :0               Invalid Argument:0
         Application Overflow :0               Device in use   :0
         Control Pipe Stall   :0               Malloc Error    :0
         Device Stalled       :0               Bad Command Code:0
         Device Detached      :0               Unknown Error   :0
         Invalid Logic Unit Num:0
USB Aladdin Token Driver Counters:
         Token Inserted       :1               Token Removed   :0
         Send Insert Msg Fail :0               Response Txns   :434
         Dev Entry Add Fail   :0               Request Txns    :434
         Dev Entry Remove Fail:0               Request Txn Fail:0
         Response Txn Fail    :0               Command Txn Fail:0
         Txn Invalid Dev Handle:0
USB Flash File System Counters:
         Flash Disconnected   :0               Flash Connected :1
         Flash Device Fail    :0               Flash Ok        :1
         Flash startstop Fail :0               Flash FS Fail   :0
USB Secure Token File System Counters:
         Token Inserted       :1               Token Detached  :0
         Token FS success     :1               Token FS Fail   :0
         Token Max Inserted   :0               Create Talker Failures:0
         Token Event          :0               Destroy Talker Failures:0
         Watched Boolean Create Failures:0
```

Displaying USB Token Infomation

Use the **dir** command with the **filesystem** keyword option **usbtoken***0-9***:** to display all files, directories, and their permission strings on the USB token.

The following sample output displays directory information for the USB token:

```
Device# dir usbtoken1:
Directory of usbtoken1:/
    2  d---          64  Dec 22 2032 05:23:40 +00:00  1000
    5  d---        4096  Dec 22 2032 05:23:40 +00:00  1001
    8  d---           0  Dec 22 2032 05:23:40 +00:00  1002
   10  d---         512  Dec 22 2032 05:23:42 +00:00  1003
   12  d---           0  Dec 22 2032 05:23:42 +00:00  5000
   13  d---           0  Dec 22 2032 05:23:42 +00:00  6000
   14  d---           0  Dec 22 2032 05:23:42 +00:00  7000
   15  ----         940  Jun 27 1992 12:50:42 +00:00  mystartup-config
   16  ----        1423  Jun 27 1992 12:51:14 +00:00  myrunning-config
32768 bytes total (858 bytes free)
```

The following sample output displays directory information for all devices to which the device is aware:

```
Device# dir all-filesystems
Directory of archive:/
No files in directory
No space information available
Directory of system:/
    2  drwx           0                    <no date>  its
  115  dr-x           0                    <no date>  lib
  144  dr-x           0                    <no date>  memory
    1  -rw-        1906                    <no date>  running-config
  114  dr-x           0                    <no date>  vfiles
No space information available
Directory of flash:/
    1  -rw-    30125020  Dec 22 2032 03:06:04 +00:00  c3825-entservicesk9-mz.123-14.T
129880064 bytes total (99753984 bytes free)
Directory of nvram:/
  476  -rw-        1947                    <no date>  startup-config
  477  ----          46                    <no date>  private-config
  478  -rw-        1947                    <no date>  underlying-config
    1  -rw-           0                    <no date>  ifIndex-table
    2  ----           4                    <no date>  rf_cold_starts
    3  ----          14                    <no date>  persistent-data
491512 bytes total (486395 bytes free)
Directory of usbflash0:/
    1  -rw-    30125020  Dec 22 2032 05:31:32 +00:00  c3825-entservicesk9-mz.123-14.T
63158272 bytes total (33033216 bytes free)
```

```
Directory of usbtoken1:/
    2  d---          64  Dec 22 2032 05:23:40 +00:00  1000
    5  d---        4096  Dec 22 2032 05:23:40 +00:00  1001
    8  d---           0  Dec 22 2032 05:23:40 +00:00  1002
   10  d---         512  Dec 22 2032 05:23:42 +00:00  1003
   12  d---           0  Dec 22 2032 05:23:42 +00:00  5000
   13  d---           0  Dec 22 2032 05:23:42 +00:00  6000
   14  d---           0  Dec 22 2032 05:23:42 +00:00  7000
   15  ----         940  Jun 27 1992 12:50:42 +00:00  mystartup-config
   16  ----        1423  Jun 27 1992 12:51:14 +00:00  myrunning-config
32768 bytes total (858 bytes free)
```

Configuration Examples for PKI Storage

Example: Storing Certificates to a Specific Local Storage Location

The following configuration example shows how to store certificates to the certs subdirectory. The certs subdirectory does not exist and is automatically created.

```
Router# dir nvram:
 114  -rw-        4687                    <no date>  startup-config
 115  ----        5545                    <no date>  private-config
 116  -rw-        4687                    <no date>  underlying-config
   1  ----          34                    <no date>  persistent-data
   3  -rw-         707                    <no date>  ioscaroot#7401CA.cer
   9  -rw-         863                    <no date>  msca-root#826E.cer
  10  -rw-         759                    <no date>  msca-root#1BA8CA.cer
  11  -rw-         863                    <no date>  msca-root#75B8.cer
  24  -rw-        1149                    <no date>  storagename#6500CA.cer
  26  -rw-         863                    <no date>  msca-root#83EE.cer
129016 bytes total (92108 bytes free)
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# crypto pki certificate storage disk0:/certs
Requested directory does not exist -- created
Certificates will be stored in disk0:/certs/
Router(config)# end
Router# write
*May 27 02:09:00:%SYS-5-CONFIG_I:Configured from console by consolemem
Building configuration...
[OK]
Router# directory disk0:/certs
Directory of disk0:/certs/
  14  -rw-         707  May 27 2005 02:09:02 +00:00  ioscaroot#7401CA.cer
  15  -rw-         863  May 27 2005 02:09:02 +00:00  msca-root#826E.cer
  16  -rw-         759  May 27 2005 02:09:02 +00:00  msca-root#1BA8CA.cer
  17  -rw-         863  May 27 2005 02:09:02 +00:00  msca-root#75B8.cer
  18  -rw-        1149  May 27 2005 02:09:02 +00:00  storagename#6500CA.cer
  19  -rw-         863  May 27 2005 02:09:02 +00:00  msca-root#83EE.cer
47894528 bytes total (20934656 bytes free)
! The certificate files are now on disk0/certs:
```

Example: Logging Into a USB Token and Saving RSA Keys to the USB Token

The following configuration example shows to how log in to the USB token, generate RSA keys, and store the RSA keys on the USB token:

```
! Configure the router to automatically log into the eToken
configure terminal
 crypto pki token default user-pin 0 1234567890
! Generate RSA keys and enroll certificates with the CA.
crypto pki trustpoint IOSCA
 enrollment url http://10.23.2.2
 exit
crypto ca authenticate IOSCA
Certificate has the following attributes:
       Fingerprint MD5:23272BD4 37E3D9A4 236F7E1A F534444E
      Fingerprint SHA1:D1B4D9F8 D603249A 793B3CAF 8342E1FE 3934EB7A
% Do you accept this certificate? [yes/no]:yes
Trustpoint CA certificate accepted.
crypto pki enroll
crypto pki enroll IOSCA
%
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this
   password to the CA Administrator in order to revoke your certificate.
   For security reasons your password will not be saved in the configuration.
   Please make a note of it.
Password:
Re-enter password:
% The subject name in the certificate will include:c2851-27.cisco.com
% Include the router serial number in the subject name? [yes/no]:no
% Include an IP address in the subject name? [no]:no
Request certificate from CA? [yes/no]:yes
% Certificate request sent to Certificate Authority
% The 'show crypto ca certificate IOSCA verbose' command will show the fingerprint.
*Jan 13 06:47:19.413:CRYPTO_PKI: Certificate Request Fingerprint MD5:E6DDAB1B
 0E30EFE6 54529D8A DA787DBA
*Jan 13 06:47:19.413:CRYPTO_PKI: Certificate Request Fingerprint SHA1:3B0F33B
7 57C02A10 3935042B C4B6CD3D 61039251
*Jan 13 06:47:21.021:%PKI-6-CERTRET:Certificate received from Certificate Authority
! Issue the write memory command, which will automatically save the RSA keys to the eToken
 ! instead of private NVRAM.
Router# write memory
Building configuration...
[OK]
*Jan 13 06:47:29.481:%CRYPTO-6-TOKENSTOREKEY:Key c2851-27.cisco.com stored on
Cryptographic Token eToken Successfully
```

The following sample output from the **show crypto key mypubkey rsa** command displays stored credentials after they are successfully loaded from the USB token. Credentials that are stored on the USB token are in the protected area. When storing the credentials on the USB token, the files are stored in a directory called /keystore. However, the key files are hidden from the command-line interface (CLI).

```
Router#
show crypto key mypubkey rsa
% Key pair was generated at:06:37:26 UTC Jan 13 2005
Key name:c2851-27.cisco.com
 Usage:General Purpose Key
 Key is not exportable.
 Key Data:
  305C300D 06092A86 4886F70D 01010105 00034B00 30480241 00E3C644 43AA7DDD
  732E0F4E 3CA0CDAB 387ABF05 EB8F22F2 2431F1AE 5D51FEE3 FCDEA934 7FBD3603
  7C977854 B8E999BF 7FC93021 7F46ABF8 A4BA2ED6 172D3D09 B5020301 0001
% Key pair was generated at:06:37:27 UTC Jan 13 2005
Key name:c2851-27.cisco.com.server
 Usage:Encryption Key
 Key is not exportable.
 Key Data:
```

```
307C300D 06092A86 4886F70D 01010105 00036B00 30680261 00DD96AE 4BF912EB
2C261922 4784EF98 2E70E837 774B3778 7F7AEB2D 87F5669B BF5DDFBC F0D521A5
56AB8FDC 9911968E DE347FB0 A514A856 B30EAFF4 D1F453E1 003CFE65 0CCC6DC7
21FBE3AC 2F8DEA16 126754BC 1433DEF9 53266D33 E7338C95 BB020301 0001
```

Additional References

Related Documents

Related Topic	Document Title
Connecting the USB modules to the router	*Cisco Access Router USB Flash Module and USB eToken Hardware Installation Guide*
eToken and USB flash data sheet	*USB eToken and USB Flash Features Support*
RSA keys	Deploying RSA Keys Within a PKI
File management (loading, copying, and rebooting files)	*Cisco Configuration Fundamentals Configuration Guide* on Cisco.com
USB Token RSA Operations: Certificate server configuration	“Configuring and Managing a Cisco IOS Certificate Server for PKI Deployment” feature document. See the “Generating a Certificate Server RSA Key Pair” section, the “Configuring a Certificate Server Trustpoint” section, and related examples.
USB Token RSA Operations: Using USB tokens for RSA operations upon initial autoenrollment	See the “Configuring Certificate Enrollment or Autoenrollment” section of the “Configuring Certificate Enrollment for a PKI ” feature document.
SDP setup, configuration and use with USB tokens	See the feature information section for the feature names on using SDP and USB tokens to deploy PKI credentials in the “Setting Up Secure Device Provisioning (SDP) for Enrollment in a PKI” feature document.

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Storing PKI Credentials

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 13: Feature Information for Storing PKI Credentials

Feature Name	Releases	Feature Information
Certificate -- Storage Location Specification		This feature allows you to specify the storage location of local certificates for platforms that support storing certificates as separate files. All Cisco platforms support NVRAM, which is the default location, and flash local storage. Depending on your platform, you may have other supported local storage options including bootflash, slot, disk, USB flash, or USB token. The following commands were introduced by this feature: **crypto pki certificate storage**, **show crypto pki certificates storage**.
RSA 4096-bit Key Generation in Software Crypto Engine Support	15.1(1)T	The range value for the **modulus** keyword value for the **crypto key generate rsa** command is extended from 360 to 2048 bits to 360 to 4096 bits.

CHAPTER 10

Source Interface Selection for Outgoing Traffic with Certificate Authority

The Source Interface Selection for Outgoing Traffic with Certificate Authority feature allows you to specify that the address of an interface be used as the source address for all outgoing TCP connections associated with that trustpoint when a designated trustpoint has been configured.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Source Interface Selection for Outgoing Traffic with Certificate Authority

Certificates That Identify an Entity

Certificates can be used to identify an entity. A trusted server, known as the certification authority (CA), issues the certificate to the entity after determining the identity of the entity. A router that is running Cisco IOS XE software obtains its certificate by making a network connection to the CA. Using the Simple Certificate Enrollment Protocol (SCEP), the router transmits its certificate request to the CA and receives the granted certificate. The router obtains the certificate of the CA in the same manner using SCEP. When validating a certificate from a remote device, the router may again contact the CA or a Lightweight Directory Access Protocol (LDAP) or HTTP server to determine whether the certificate of the remote device has been revoked. (This process is known as checking the certificate revocation list [CRL].)

Note Depending on your Cisco IOS release, LDAP is supported.

In some configurations, the router may make the outgoing TCP connection using an interface that does not have a valid or routable IP address. The user must specify that the address of a different interface be used as the source IP address for the outgoing connection. Cable modems are a specific example of this requirement because the outgoing cable interface (the RF interface) usually does not have a routable address. However, the user interface (usually FastEthernet) does have a valid IP address.

Source Interface for Outgoing TCP Connections Associated with a Trustpoint

The **crypto pki trustpoint** command is used to specify a trustpoint. The **source interface**command is used along with the **crypto pki trustpoint**command to specify the address of the interface that is to be used as the source address for all outgoing TCP connections associated with that trustpoint.

Note If the interface address is not specified using the **source interface**command, the address of the outgoing interface is used.

How to Configure Source Interface Selection for Outgoing Traffic with Certificate Authority

Configuring the Interface for All Outgoing TCP Connections Associated with a Trustpoint

Perform this task to configure the interface that you want to use as the source address for all outgoing TCP connections associated with a trustpoint.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment url** *url*
5. **source interface** *interface-address*
6. **interface** *type slot* / *port*
7. **description** *string*
8. **ip address** *ip-address mask*
9. **interface** *type slot* / *port*
10. **description** *string*
11. **ip address** *ip-address mask*
12. **crypto map** *map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Router (config)# crypto pki trustpoint ms-ca`	Declares the Certificate Authority (CA) that your router should use and enters ca-trustpoint configuration mode.
Step 4	**enrollment url** *url* **Example:**	Specifies the enrollment parameters of your CA.

	Command or Action	Purpose
	`Router (ca-trustpoint)# enrollment url http://yourname:80/certsrv/mscep/mscep.dll`	
Step 5	**source interface** *interface-address* **Example:** `Router (ca-trustpoint)# interface fastethernet1/0`	Interface to be used as the source address for all outgoing TCP connections associated with that trustpoint.
Step 6	**interface** *type slot* / *port* **Example:** `Router (ca-trustpoint)# interface fastethernet1/0`	Configures an interface type and enters interface configuration mode.
Step 7	**description** *string* **Example:** `Router (config-if)# description inside interface`	Adds a description to an interface configuration.
Step 8	**ip address** *ip-address mask* **Example:** `Router (config-if)# ip address 10.1.1.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 9	**interface** *type slot* / *port* **Example:** `Router (config-if)# interface fastethernet1/0`	Configures an interface type.
Step 10	**description** *string* **Example:** `Router (config-if)# description outside interface 10.1.1.205 255.255.255.0`	Adds a description to an interface configuration.
Step 11	**ip address** *ip-address mask* **Example:** `Router (config-if)# ip address 10.2.2.205 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 12	**crypto map** *map-name* **Example:** `Router (config-if)# crypto map mymap`	Applies a previously defined crypto map set to an interface.

Troubleshooting Tips

Ensure that the interface specified in the command has a valid address. Attempt to ping the router using the address of the specified interface from another device (possibly the HTTP or LDAP server that is serving the CRL). You can do the same thing by using a traceroute to the router from the external device.

You can also test connectivity between the router and the CA or LDAP server by using Cisco IOS XE command-line interface (CLI). Enter the **ping ip**command and respond to the prompts. If you answer "yes" to the "Extended commands [n]:" prompt, you will be able to specify the source address or interface.

In addition, you can use Cisco IOS XE CLI to input a traceroute command. If you enter the **traceroute ip** command (in EXEC mode), you will be prompted for the destination and source address. You should specify the CA or LDAP server as the destination and the address of the interface that you specified in the "source interface" as the source address.

Configuration Examples for Source Interface Selection for Outgoing Traffic with Certificate Authority

Source Interface Selection for Outgoing Traffic with Certificate Authority Example

In the following example, the router is located in a branch office. The router uses IP Security (IPSec) to communicate with the main office. FastEthernet 1 is the "outside" interface that connects to the Internet Service Provider (ISP). FastEthernet 0 is the interface connected to the LAN of the branch office. To access the CA server located in the main office, the router must send its IP datagrams out interface FastEthernet 1 (address 10.2.2.205) using the IPSec tunnel. Address 10.2.2.205 is assigned by the ISP. Address 10.2.2.205 is not a part of the branch office or main office.

The CA cannot access any address outside the company because of a firewall. The CA sees a message coming from 10.2.2.205 and cannot respond (that is, the CA does not know that the router is located in a branch office at address 10.1.1.1, which it is able to reach).

Adding the **source interface** command tells the router to use address 10.1.1.1 as the source address of the IP datagram that it sends to the CA. The CA is able to respond to 10.1.1.1.

This scenario is configured using the **source interface** command and the interface addresses as described above.

```
crypto pki trustpoint ms-ca
 enrollment url http://ms-ca:80/certsrv/mscep/mscep.dll
 source interface fastethernet0
!
interface fastethernet 0
 description inside interface
 ip address 10.1.1.1 255.255.255.0
!
interface fastethernet 1
 description outside interface
 ip address 10.2.2.205 255.255.255.0
 crypto map main-office
```

Additional References

The following sections provide references related to the Source Interface Selection for Outgoing Traffic with Certificate Authority feature.

Related Documents

Related Topic	Document Title
Configuring IPSec and certification authority	Security for VPNs with IPsec
IPSec and certification authority commands	*Cisco IOS Security Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature.	-

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature.	-

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Source Interface Selection for Outgoing Traffic with Certificate Authority

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for Source Interface Selection for Outgoing Traffic with Certificate Authority

Feature Name	Releases	Feature Information
Source Interface Selection for Outgoing Traffic with Certificate Authority.	Cisco IOS XE Release 2.1	This feature allows you to specify that the address of an interface be used as the source address for all outgoing TCP connections associated with that trustpoint when a designated trustpoint has been configured. The following command was introduced: **source interface**.

Glossary

authenticate--Toprove the identity of an entity using the certificate of an identity and a secret that the identity poses (usually the private key corresponding to the public key in the certificate).

CA --Certificate Authority. A CA is an entity that issues digital certificates (especially X.509 certificates) and vouches for the binding between the data items in a certificate.

CA authentication --The user manually approves a certificate from a root CA. Usually a fingerprint of the certificate is presented to the user, and the user is asked to accept the certificate based on the fingerprint. The certificate of a root CA is signed by itself (self-signed) so that it cannot be automatically authenticated using the normal certificate verification process.

CRL --certificate revocation list. A CRL is a data structure that enumerates digital certificates that have been invalidated by their issuer prior to when they were scheduled to expire.

enrollment --A router receives its certificate via the enrollment process. The router generates a request for a certificate in a specific format (known as PKCS #10). The request is transmitted to a CA, which grants the request and generates a certificate encoded in the same format as the request. The router receives the granted certificate and stores it in an internal database for use during normal operations.

certificate --A data structure defined in International Organization for Standardization (ISO) standard X.509 to associate an entity (machine or human) with the public key of that entity. The certificate contains specific fields, including the name of the entity. The certificate is normally issued by a CA on behalf of the entity. In this case the router will act as its own CA. Common fields within a certificate include the distinguished name (DN) of the entity, the DN of the authority issuing the certificate, and the public key of the entity.

LDAP --Lightweight Directory Access Protocol. A LDAP is a protocol that provides access for management and browser applications that provide read-and-write interactive access to the X.500 directory.

CHAPTER 11

PKI Trustpool Management

The PKI Trustpool Management feature is used to authenticate sessions, such as HTTPS, that occur between devices by using commonly recognized trusted agents called certificate authorities (CAs).

Trustpool certificates are well-known CA certificates with which you can establish trust. IOS PKI has both built-in CAs and also has an option to download trustpool bundle. Built-in CA certificates are used to verify PKCS7 signature of downloaded trustpool bundle. You can download the trustpool bundle if signature verification fails. You can delete Built-in trustpool certificates. Trustpool certificates are used by applications such as SSLVPN, PnP, Smart License, MacSec and so on.

This feature, which is enabled by default, is used to create a scheme to provision, store, and manage a pool of certificates from known CAs in a way similar to the services a browser provides for securing sessions.

Note A new root certificate is included in the built-in certificates for Cisco Plug and Play application.

Note Effective with Cisco IOS XE Denali 16.3, the way PKI Trustpools are managed have changed. If you are planning to upgrade to this release, please review the changes to the feature captured below as part of *PKI Trustpool Enhancements* section.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for PKI Trustpool Management

The use of certificates requires that a crypto subsystem is included in the Cisco IOS software image.

Restrictions for PKI Trustpool Management

Device certificates that use CA certificates cannot be enrolled in a PKI trustpool.

You can download only a Cisco signed PKCS7 certificate through the trustpool URL.

Information About PKI Trustpool Management

CA Certificate Storage in a PKI Trustpool

The router uses a built-in CA certificate bundle that is contained in a special certificate store called a PKI trustpool, which is updated automatically from Cisco. This PKI trustpool is known by Cisco and other vendors. A CA certificate bundle can be in the following formats:

- X.509 certificates in Distinguished Encoding Rules (DER) binary format enveloped within a public-key cryptographic message syntax standard 7 (pkcs7), which is used to sign and encrypt messages under a PKI. An X.509 certificate is a PKI and Privilege Management Infrastructure (PMI) standard that specifies, among other things, standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm.
- A file containing concatenated X.509 certificates in Privacy Enhanced Mail (PEM) format with PEM headers.

Note Flash can also be used as the storage location for the bundles instead of NVRAM.

PKI Trustpool Updating

The PKI trustpool is treated as a single entity that needs to be updated when the following conditions occur:

- A certificate in the PKI trustpool is due to expire or has been reissued.
- The published CA certificate bundle contains additional trusted certificates that are needed by a given application.
- The configuration has been corrupted.

Note A built-in certificate in the PKI trustpool cannot be physically replaced. However, a built-in certificate is rendered inactive after an update if its X.509 subject-name attribute matches the certificate in the CA certificate bundle.

The PKI trustpool can be updated automatically or manually. The PKI trustpool may be used by certficate validation depending upon the application using it. See the "Manually Updating Certificates in the PKI Trustpool" and "Configuring Optional PKI Trustpool Policy Parameters" sections for more information.

Note During auto-update all the existing downloaded trustpool certificates must be deleted.

The PKI trustpool timer matches the CA certificate with the earliest expiration time. If the timer is running and a bundle location is not configured and not explicitly disabled, syslog warnings are issued to alert the administrator that the PKI trustpool policy option is not set.

Automatic PKI trustpool updates use the configured URL.

When the PKI trustpool expires, the policy is read, the bundle is loaded, and the PKI trustpool is replaced. If the automatic PKI trustpool update encounters problems when initiating, then the following schedule is used to initiate the update until the download is successful: 20 days, 15 days, 10 days, 5 days, 4 days, 3 days, 2 days, 1 day, and then once every hour.

CA Handling in Both PKI Trustpool and Trustpoint

There may be circumstances where a CA resides in both PKI trustpool and trustpoint; for example, a trustpoint uses a CA and a CA bundle is downloaded later with the same CA inside. In this scenario, the CA in the trustpoint and the policy of this trustpoint is considered before the CA in the PKI trustpool or PKI trustpool policy to ensure that any current behavior is not altered when the PKI Trustpool Management feature is implemented on the router.

PKI Trustpool Enhancements

In releases earlier than Cisco IOS XE Denali 16.3, the trustpool consists of built-in certificates deployed with every Cisco box and downloaded CA certificates from published bundles. The downloaded certificates are saved in NVRAM, by default. The certificates from the downloaded trustpool bundle would be extracted and stored in the running configuration which was inefficient and utilized too much space.

From Cisco IOS XE Denali 16.3, the PKI trustpool enhancements stores the bundles in the same downloaded bundle format as one file in the storage location (default is NVRAM) instead of individual certificates like in the previous releases. This helps in saving storage memory as the file is in compressed format. Also, the certificates are not displayed individually in the running configuration. On every reboot the bundles are read from the storage location and individual certificates are installed in the database.

This feature removes the current downloaded certificates from the running configuration. The **crypto pki certificate pool** will not have the DER format certificates because these certificates are incompatible with the old NVRAM file and the new images. During upgrade, the trustpool certificates in DER format are lost and the bundles must be reinstalled again in the storage. This is indicated by a syslog during reboot in case of old NVRAM files. The **show crypto pki trustpool** command indicates that the configuration has been removed. Before you upgrade, use the **show crypto pki trustpool** command to verify that the certificates are available.

The following steps must be followed before upgrading to Cisco IOS XE Denali 16.3 :

- Remove the downloaded trustpool certificates using the **crypto pki trustpool clean** command
- Use the **write memory** command
- Reboot the device
- Download the trustpool bundles using the **crypto pki trustpool import url** command

If you are using trustpool certificates to log into SSH, then you need to follow additional steps to transfer that specific certificate from bundle to a trustpoint. See *Example: Using PKI Trustpool for SSH Connection During Upgrade* for more information.

Note From Cisco IOS XE Gibraltar 16.10 release onwards, when you configure the **match crlsign** command under trustpoint, the crlsign will be crosss checked while validating.

How to Configure PKI Trustpool Management

Manually Updating Certificates in the PKI Trustpool

The PKI Trustpool Management feature is enabled by default and uses the built-in CA certificate bundle in the PKI trustpool, which receives automatic updates from Cisco. Perform this task to manually update certificates in the PKI trustpool if they are not current, are corrupt, or if certain certificates need to be updated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpool import clean** [**terminal** | **url** *url*]
4. **crypto pki trustpool import** {**terminal**} {**url** *url* | **ca-bundle**} {**vrf** *vrf-name* | **source interface** *interface-name*}
5. **exit**
6. **show crypto pki trustpool**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpool import clean** [**terminal** | **url** *url*] **Example:** `Device(config)# crypto pki trustpool import clean`	(Optional) Manually removes all downloaded PKI CA certificates. - The **clean** keyword specifies the removal of the downloaded PKI trustpool certificates before the new certificates are downloaded. - The **terminal** keyword removes the existing CA certificate bundle terminal setting. - The **url** keyword and *url* argument removes the existing URL file system setting.
Step 4	**crypto pki trustpool import** {**terminal**} {**url** *url* | **ca-bundle**} {**vrf** *vrf-name* | **source interface** *interface-name*} **Example:** `Device(config)# crypto pki trustpool import url http://www.cisco.com/security/pki/trs/ios.p7b`	Manually imports (downloads) the CA certificate bundle into the PKI trustpool to update or replace the existing CA certificate bundle. - The **terminal** keyword specifies the importation of a CA certificate bundle through the terminal (cut-and-paste) in PEM format. - The **url** keyword with the *url* argument specifies the importation of a CA certificate bundle through a URL. This URL can be through a variety of URL file systems such as HTTP. See the *PKI Trustpool Updating* section for more information. In CA bundle, you can use the **crypto pki trustpool import** command to pass the traffic through global VRF. Also, the traffic will not divert through a VRF, when you configure the **crypto pki trustpool policy** command specifying the VRF and source interface.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.
Step 6	**show crypto pki trustpool** **Example:** `Device(config)# show crypto pki trustpool`	Displays the PKI trustpool certificates of the router in a verbose format.

Configuring Optional PKI Trustpool Policy Parameters

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpool policy**
4. **cabundle url** {*url* | **none**}
5. **chain-validation**
6. **crl** {**cache** {**delete-after** {*minutes* | **none**} | **query** *url*}
7. **default** *command-name*
8. **match certificate** *certificate-map-name* [**allow expired-certificate** | **override** {**cdp directory** *ldap-location* | **ocsp** {*number* **url** *url* | **trustpool** *name number* **url** *url*} | **sia** *number url*} | **skip** [**revocation-check** | **authorization-check**]]
9. **ocsp** {**disable-nonce** | **url** *url*}
10. **revocation-check** *method1* [*method2* [*method3*]]
11. **source interface** *name number*
12. **storage** *location*
13. **vrf** *vrf-name*
14. **show**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**crypto pki trustpool policy**
	Example:
	`Device(config)# crypto pki trustpool policy`
	`Device(ca-trustpool)#`
		Enters ca-trustpool configuration mode where commands can be accessed to configure CA PKI trustpool policy parameters. The trustpool policy only affects the crl retrieval process and has no effect on trustpool import process.

Step 4	**cabundle url** {*url* | **none**}
	Example:
	`Device(ca-trustpool)# cabundle url`
	`http://www.cisco.com/security/pki/crl/crca2048.crl`
		Specifies the URL from which the PKI trustpool certificate authority CA certificate bundle is downloaded .
		• The *url* argument is the URL of the CA certificate bundle.
		• The **none** keyword specifies that autoupdates of the PKI trustpool CA are not permitted.

	Command or Action	Purpose
Step 5	**chain-validation** **Example:** `Device(ca-trustpool)# chain-validation`	Enables chain validation from the peer's certificate to the root CA certificate in the PKI trustpool. The default has validation stopping at the peer certificate's issuer.

Step 6	**crl** {**cache** {**delete-after** {*minutes* | **none**} | **query** *url*}
	Example:
	`Device(ca-trustpool)# crl query http://www.cisco.com/security/pki/crl/crca2048.crl`
		Specifies the certificate revocation list (CRL) query and CRL cache options for the PKI trustpool.
		• The **cache** keyword specifies CRL cache options.
		• The **delete-after** keyword removes the CRL from the cache after a timeout.
		• The *minutes* argument is the number of minutes from 1 to 43,200 to wait before deleting the CRL from the cache.
		• The **none** keyword specifies that CRLs are not cached.
		• The **query** keyword with the *url* argument specifies the URL published by the CA server to query the CRL.

Step 7	**default** *command-name* **Example:** `Device(ca-trustpool)# default crl query http://www.cisco.com/security/pki/crl/crca2048.crl`	Resets the value of a ca-trustpool configuration subcommand to its default . • The *command-name* argument is the ca-trustpool configuration mode command with its applicable keywords.
Step 8	**match certificate** *certificate-map-name* [**allow expired-certificate** | **override** {**cdp directory** *ldap-location* | **ocsp** {*number* **url** *url* | **trustpool** *name number* **url** *url*} | **sia** *number url*} | **skip** [**revocation-check** | **authorization-check**]] **Example:** `match certificate mycert override ocsp 1 url http://ocspts.identrust.com`	Enables the use of certificate maps for the PKI trustpool. • The *certifcate-map-name* argument matches the certificate map name. • The optional **allow expired-certificate** keyword ignores expired certificates. **Note** If this keyword is not configured, the router does not ignore expired certificates. • The **override** keyword overrides the online certificate status protocol (OCSP) or SubjectInfoAccess (SIA) attribute fields in a certificate that is in the PKI trustpool. • The **cdp** keyword overrides the certificate distribution point (CDP) in a certificate.

	Command or Action	Purpose
		- The **directory** keyword and *ldap-location* specifies the CDP in either the http: or ldap: URL, or LDAP directory to override in the certificate. - The **ocsp** keyword and *number* argument and **url** keyword and *url* argument specifies the OCSP sequence number from 0 to 10000 and URL to override in the certificate. - The **trustpool** keyword and *name* and *number* arguments with the **url** keyword and *url* argument override the PKI trustpool for verifying the OCSP certificate by specifying the PKI trustpool name, sequence number, and URL. - The **sia** keyword and *number* and *url* arguments override the SIA URL in a certificate by specifying the SIA sequence number and URL. - The optional **skip revocation-check** keyword combination allows the PKI trustpool to enforce certificate revocation lists (CRLs) except for specific certificates. **Note** If this keyword combination is not configured, then the PKI trustpool enforces CRLs for all certificates. - The optional **skip authorization-check** keyword combination skips the authentication, authorization, and accounting (AAA) check of a certificate when public key infrastructure (PKI) integration with an AAA server is configured. **Note** If this keyword combination is not configured, and PKI integration with an AAA server is configured, then the AAA checking of a certificate is done.

Step 9	**ocsp** {**disable-nonce** | **url** *url*}
	Example:
	`Device(ca-trustpool)# ocsp url http://ocspts.identrust.com`
		Specifies OCSP settings for the PKI trustpool.
		- The **disable-nonce** keyword disables the OCSP Nonce extension.
		- The **url** keyword and *url* argument specify the OCSP server URL to override (if one exists) in the Authority Info Access (AIA) extension of the certificate. All certificates associated with a configured PKI trustpool are checked by the OCSP server at the specified HTTP URL. The URL can be a hostname, IPv4 address, or an IPv6 address.

	Command or Action	Purpose
Step 10	**revocation-check** *method1* [*method2* [*method3*]] **Example:** `Device(ca-trustpool)# revocation-check ocsp crl none`	Disables revocation checking when the PKI trustpool policy is being used. The *method* argument is used by the router to check the revocation status of the certificate. Available keywords are as follows: • The **crl** keyword performs certificate checking by a certificate revocation list (CRL). This is the default behavior. • The **none** keyword does not require a certificate checking. • The **ocsp** keyword performs certificate checking by an online certificate status protocol (OCSP) server. If a second and third method are specified, each method is used only if the previous method returns an error, such as a server being down.
Step 11	**source interface** *name number* **Example:** `Device(ca-trustpool)# source interface tunnel 1`	Specifies the source interface to be used for CRL retrieval, OCSP status, or the downloading of a CA certificate bundle for the PKI trustpool . • The *name* and *number*arguments are for the interface type and number used as the source address for the PKI trustpool.
Step 12	**storage** *location* **Example:** `Device(ca-trustpool)# storage storage disk0:crca2048.crl`	Specifies a file system location where PKI trustpool certificates are stored on the router. • The *location* is the file system location where the PKI trustpool certificates are stored. The types of file system locations are **disk0:**, **disk1:**, **nvram:**, **unix:**, or a named file system.
Step 13	**vrf** *vrf-name* **Example:** `Device(ca-trustpool)# vrf myvrf`	Specifies the VPN routing and forwarding (VRF) instance to be used for enrolment, CRL retrieval, and OCSP status.

Step 14	**show**
	Example:
	`Device(ca-trustpool)# show`
	`Chain validation will stop at the first CA certificate in the pool`
	`  Trustpool CA certificates will expire 12:58:31 PST Apr 5 2012`
	`  Trustpool policy revocation order:      crl`
	`  Certficate matching is disabled`
	`  Policy Overrides:`
		Displays the PKI trustpool policy of the router.

Configuration examples for PKI Trustpool Management

Example: Configuring PKI Trustpool Management

The following **show crypto pki trustpool** command output displays the certificates in PKI trustpool:

Note The command output in this example is abridged because it is verbose.

```
Device# show crypto pki trustpool

CA Certificate
  Status: Available
  Version: 3
  Certificate Serial Number (hex): 00D01E474000000111C38A964400000002
  Certificate Usage: Signature
  Issuer:
    cn=DST Root CA X3
    o=Digital Signature Trust Co.
  Subject:
    cn=Cisco SSCA
    o=Cisco Systems
  CRL Distribution Points:
    http://crl.identrust.com/DSTROOTCAX3.crl
  Validity Date:
    start date: 12:58:31 PST Apr 5 2007
    end   date: 12:58:31 PST Apr 5 2012

CA Certificate
  Status: Available
  Version: 3
  Certificate Serial Number (hex): 6A6967B3000000000003
  Certificate Usage: Signature
  Issuer:
    cn=Cisco Root CA 2048
    o=Cisco Systems
  Subject:
    cn=Cisco Manufacturing CA
    o=Cisco Systems
  CRL Distribution Points:
    http://www.cisco.com/security/pki/crl/crca2048.crl
  Validity Date:
    start date: 14:16:01 PST Jun 10 2005
    end   date: 12:25:42 PST May 14 2029
```

The following **show crypto pki trustpool verbose** command output displays the certificates in PKI trustpool:

```
Device# show crypto pki trustpool verbose

CA Certificate
  Status: Available
  Version: 3
  Certificate Serial Number (hex): 01
  Certificate Usage: Signature
  Issuer:
    cn=Licensing Root - DEV
    o=Cisco
  Subject:
    cn=Licensing Root - DEV
    o=Cisco
  Validity Date:
    start date: 03:25:43 IST Apr 25 2013
    end   date: 03:25:43 IST Apr 25 2033
  Subject Key Info:
    Public Key Algorithm: rsaEncryption
    RSA Public Key: (2048 bit)
  Signature Algorithm: SHA256 with RSA Encryption
  Fingerprint MD5: 432CBFA0 32D2983A 8A56A319 FD28C6F9
  Fingerprint SHA1: 6341FCAF 19CE9FEE 961D92A5 D47390B5 2DD6D94D
  X509v3 extensions:
    X509v3 Key Usage: 6000000
      Key Cert Sign
      CRL Signature
    X509v3 Subject Key ID: 43214521 B5FB217A 1A4D1BB7 0236E664 CBEC8B65
    X509v3 Basic Constraints:
        CA: TRUE
    Authority Info Access:
  Associated Trustpoints: Trustpool
  Trustpool: Built-In
```

Example: Using PKI Trustpool for SSH Connection During Upgrade

Before upgrading to Cisco IOS XE Denali 16.3, copy the certificate from trustpool to a new trustpoint.

```
Device # show run | sec pool
crypto pki trustpool policy
 revocation-check none
 source interface GigabitEthernet0/0/0
crypto pki certificate pool
 certificate ca 01
  308204FA 308202E2 A0030201 02020101 300D0609 2A864886 F70D0101 0C050030
  0E310C30 0A060355 04031303 61626330 1E170D31 36303730 35303435 3935335A
  170D3136 30373035 30353535 35335A30 0E310C30 0A060355 04031303 61626330
  82022230 0D06092A 864886F7 0D010101 05000382 020F0030 82020A02 82020100
  C78AA144 8EC1D18A 4EECC3E8 81450CC7 A85A4C57 AF59E584 5C1EA888 6EF70DA8
  33327D93 E1F6CED7 32BB4FCF 693F60E0 37000225 40F6F9C5 0462C4AD 899E5BDD
  ED779180 D6C75E1B FBE97D42 E2A7B35D DDC18C4D 4CCDE401 68F67A6D E40FD744
  904EE49F 40820640 C6E0B072 510BC40E A0883F6C E8DF5128 EFF3B5F4 B31E5C16
  217652FF AFC30EBF 593CB19C 56C0E793 2814D504 0E079E0C 8E9E856A BCADB19C
  F2376994 A0A040C1 7BC1E88F CF80F218 9C48B4D9 F84ED5C0 79827BD1 32448478
  8F1F82F2 C91A9479 692B6456 C53CF937 777D0C31 1B8A1F5E 24B33553 047C2448
  855CF974 DFA21665 8AD8A0E5 81ED8068 81688997 FF05118C 93A59CA0 7FD594F6
  B7B1898C 272E089A 3392A2C4 22A22625 2BC1E16F 95B2FC15 207CCA49 378AD3A6
  0C574197 C5E94D8C E6736271 CE0BA9AB ACB380E3 A8084243 4E038DD1 8E86E206
  E2269290 F1AFB29A D28CFB3A 5ABADE4A 21A59728 7174E7A3 2FF59C90 E6100C6E
  E2E8CB4C 91BD574D 57B5E18A 78F9CE75 624C4A2E 1A6EFCC3 7D1BB20B 1CC79024
  CD2FBC4D 46BE1B7A 6EFD8F05 6FD84E91 51215E9B E5E952A4 6E2D1388 10075706
  7D6FAF9B 3F7F8994 F39B9B5D 0C7CD5BC 40738877 5D9985AC 5AB6363D 811BA440
  41A1639F 352F4F01 1994300A A4B85B75 01486CA0 4C4B3175 82038B26 BEFE1D2A
  4AC0D577 7784FACF A6877D68 5D73DD04 DC8D942B DE3FC9FE 4C1FF715 A2E7A5AB
  02030100 01A36330 61300F06 03551D13 0101FF04 05300301 01FF300E 0603551D
  0F0101FF 04040302 0186301F 0603551D 23041830 168014CA 195EDBF1 51753A92
  71342CA8 36DDABA9 63A93130 1D060355 1D0E0416 0414CA19 5EDBF151 753A9271
  342CA836 DDABA963 A931300D 06092A86 4886F70D 01010C05 00038202 0100553B
  FB77A348 C4447C40 BEB2DDFD 63C82441 3CBDC198 B5D5B1AB DF17C4E2 98AEAF2F
  CD570939 BCC116E0 33CFF471 E91EE308 8B29B5BD 11DFACF9 A3AC3135 8BE81B22
  ED205587 5DE04654 A051CC14 CA8D2A6E 81F924DA 001BB1C4 7F85F177 4E75D8EA
  797CCAEF 1502492D 17627CD1 E39E295B 44C55884 8E6DFF68 2129B222 18E3187D
  AB97B4A7 6F838E75 A8908566 AD9E6687 35B150DE 0C8C1B37 6F17FDAC 7A7C53A4
  434F5CF3 6EB71957 E65EC5D2 7685B05B A9D8C0D3 2DB8F97E E6B37E11 C9E26F4F
  BFB97745 83E1A214 461B0E49 0FFDEF21 A7CA5364 44416002 03A01F0C 2BC098D3
  B50A4071 AC4D2234 4E55C5D4 0FD9C308 63F2A8D4 24D34613 B73EAA1B B407D56F
  90EEF5C7 AE61C0D8 13FB493D 0E1C8F9B 1D2D6DEA 458CDE18 8753FF14 F8C75213
  35557FCC 50405056 D9790AF0 EAC21646 2D9AF88D 59C05434 45F21248 0BB72191
  74D951DD 9D23997E 1134611E 837137E6 C40C694E 7AB4A05F E8470E87 E0F6D924
  A69A98A8 5AA2B9B3 B7446883 94A7230D EE3C6EDA 4A348351 FC40C16D 6FDC91EC
  CEFF580B F7826DD1 1D1D07DB 17CA3298 8C510826 D2712E04 EB669909 3D8106EB
  5391A5BA 80B7E981 B41AAEB9 CE4A5236 20E30AE7 01D5FDB3 604C5505 0F8C96DC
  8F5CF569 5D90C1FB F5679221 B7B922C0 5F11C379 9EBA283C 45A209F7 132B8DA2
  EAF4751B 290A1CAC C3E7978B 760FB05A 185991FE 4884FA1A D3EEDD7C 63
  3B
        quit
```

Paste the certificate in config mode by creating a new trustpoint.

```
Device(config)#cry pki trust abc
Device(ca-trustpoint)#cry pki cert chain abc
Device(config-cert-chain)#certificate ca 01
```

Enter the certificate in hexadecimal representation

```
Device(config-pki-hexmode)#  308204FA 308202E2 A0030201 02020101 300D0609 2A864886 F70D0101
 0C050030
Device(config-pki-hexmode)#  0E310C30 0A060355 04031303 61626330 1E170D31 36303730 35303435
 3935335A
Device(config-pki-hexmode)#  170D3136 30373035 30353535 35335A30 0E310C30 0A060355 04031303
 61626330
Device(config-pki-hexmode)#  82022230 0D06092A 864886F7 0D010101 05000382 020F0030 82020A02
 82020100
Device(config-pki-hexmode)#  C78AA144 8EC1D18A 4EECC3E8 81450CC7 A85A4C57 AF59E584 5C1EA888
 6EF70DA8
Device(config-pki-hexmode)#  33327D93 E1F6CED7 32BB4FCF 693F60E0 37000225 40F6F9C5 0462C4AD
 899E5BDD
Device(config-pki-hexmode)#  ED779180 D6C75E1B FBE97D42 E2A7B35D DDC18C4D 4CCDE401 68F67A6D
 E40FD744
Device(config-pki-hexmode)#  904EE49F 40820640 C6E0B072 510BC40E A0883F6C E8DF5128 EFF3B5F4
 B31E5C16
Device(config-pki-hexmode)#  217652FF AFC30EBF 593CB19C 56C0E793 2814D504 0E079E0C 8E9E856A
 BCADB19C
Device(config-pki-hexmode)#  F2376994 A0A040C1 7BC1E88F CF80F218 9C48B4D9 F84ED5C0 79827BD1
 32448478
Device(config-pki-hexmode)#  8F1F82F2 C91A9479 692B6456 C53CF937 777D0C31 1B8A1F5E 24B33553
 047C2448
Device(config-pki-hexmode)#  855CF974 DFA21665 8AD8A0E5 81ED8068 81688997 FF05118C 93A59CA0
 7FD594F6
Device(config-pki-hexmode)#  B7B1898C 272E089A 3392A2C4 22A22625 2BC1E16F 95B2FC15 207CCA49
 378AD3A6
Device(config-pki-hexmode)#  0C574197 C5E94D8C E6736271 CE0BA9AB ACB380E3 A8084243 4E038DD1
```

```
 8E86E206
Device(config-pki-hexmode)#  E2269290 F1AFB29A D28CFB3A 5ABADE4A 21A59728 7174E7A3 2FF59C90
 E6100C6E
Device(config-pki-hexmode)#  E2E8CB4C 91BD574D 57B5E18A 78F9CE75 624C4A2E 1A6EFCC3 7D1BB20B
 1CC79024
Device(config-pki-hexmode)#  CD2FBC4D 46BE1B7A 6EFD8F05 6FD84E91 51215E9B E5E952A4 6E2D1388
 10075706
Device(config-pki-hexmode)#  7D6FAF9B 3F7F8994 F39B9B5D 0C7CD5BC 40738877 5D9985AC 5AB6363D
 811BA440
Device(config-pki-hexmode)#  41A1639F 352F4F01 1994300A A4B85B75 01486CA0 4C4B3175 82038B26
 BEFE1D2A
Device(config-pki-hexmode)#  4AC0D577 7784FACF A6877D68 5D73DD04 DC8D942B DE3FC9FE 4C1FF715
 A2E7A5AB
Device(config-pki-hexmode)#  02030100 01A36330 61300F06 03551D13 0101FF04 05300301 01FF300E
 0603551D
Device(config-pki-hexmode)#  0F0101FF 04040302 0186301F 0603551D 23041830 168014CA 195EDBF1
 51753A92
Device(config-pki-hexmode)#  71342CA8 36DDABA9 63A93130 1D060355 1D0E0416 0414CA19 5EDBF151
 753A9271
Device(config-pki-hexmode)#  342CA836 DDABA963 A931300D 06092A86 4886F70D 01010C05 00038202
 0100553B
Device(config-pki-hexmode)#  FB77A348 C4447C40 BEB2DDFD 63C82441 3CBDC198 B5D5B1AB DF17C4E2
 98AEAF2F
Device(config-pki-hexmode)#  CD570939 BCC116E0 33CFF471 E91EE308 8B29B5BD 11DFACF9 A3AC3135
 8BE81B22
Device(config-pki-hexmode)#  ED205587 5DE04654 A051CC14 CA8D2A6E 81F924DA 001BB1C4 7F85F177
 4E75D8EA
Device(config-pki-hexmode)#  797CCAEF 1502492D 17627CD1 E39E295B 44C55884 8E6DFF68 2129B222
 18E3187D
Device(config-pki-hexmode)#  AB97B4A7 6F838E75 A8908566 AD9E6687 35B150DE 0C8C1B37 6F17FDAC
 7A7C53A4
Device(config-pki-hexmode)#  434F5CF3 6EB71957 E65EC5D2 7685B05B A9D8C0D3 2DB8F97E E6B37E11
 C9E26F4F
Device(config-pki-hexmode)#  BFB97745 83E1A214 461B0E49 0FFDEF21 A7CA5364 44416002 03A01F0C
 2BC098D3
Device(config-pki-hexmode)#  B50A4071 AC4D2234 4E55C5D4 0FD9C308 63F2A8D4 24D34613 B73EAA1B
 B407D56F
Device(config-pki-hexmode)#  90EEF5C7 AE61C0D8 13FB493D 0E1C8F9B 1D2D6DEA 458CDE18 8753FF14
 F8C75213
Device(config-pki-hexmode)#  35557FCC 50405056 D9790AF0 EAC21646 2D9AF88D 59C05434 45F21248
 0BB72191
Device(config-pki-hexmode)#  74D951DD 9D23997E 1134611E 837137E6 C40C694E 7AB4A05F E8470E87
 E0F6D924
Device(config-pki-hexmode)#  A69A98A8 5AA2B9B3 B7446883 94A7230D EE3C6EDA 4A348351 FC40C16D
 6FDC91EC
Device(config-pki-hexmode)#  CEFF580B F7826DD1 1D1D07DB 17CA3298 8C510826 D2712E04 EB669909
 3D8106EB
Device(config-pki-hexmode)#  5391A5BA 80B7E981 B41AAEB9 CE4A5236 20E30AE7 01D5FDB3 604C5505
 0F8C96DC
Device(config-pki-hexmode)#  8F5CF569 5D90C1FB F5679221 B7B922C0 5F11C379 9EBA283C 45A209F7
 132B8DA2
Device(config-pki-hexmode)#  EAF4751B 290A1CAC C3E7978B 760FB05A 185991FE 4884FA1A D3EEDD7C
 63
Device(config-pki-hexmode)#  3B
Device(config-pki-hexmode)#       quit
```

Now you can upgrade to Cisco IOS XE Denali 16.3. The certificate from trustpool would disappear but would still stay in trustpoint. Install the certificate in trustpool after the upgrade.

Additional References for PKI Trustpool Management

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

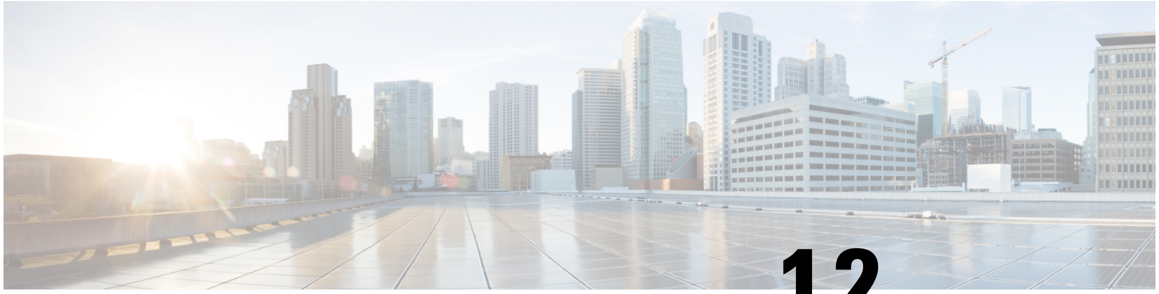
Feature Information for PKI Trustpool Management

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15: Feature Information for PKI Trustpool Management

Feature Name	Releases	Feature Information
PKI Trustpool Management		The PKI Trustpool Management feature is used to authenticate sessions, such as HTTPS, that occur between devices by using commonly recognized trusted agents called certificate authorities (CAs). The following commands were introduced or modified: **cabundle url**, **chain-validation (ca-trustpool)**, **crypto pki trustpool import**, **crypto pki trustpool policy**, **crl**, **default (ca-trustpool)**, **match certificate (ca-trustpool)**, **ocsp**, **show (ca-trustpool)**, **show crypto pki trustpool**, **source interface (ca-trustpool)**, **storage**, **vrf (ca-trustpool)**, **show crypto pki trustpool built-in**, **crypto pki trustpool import clean ca-bundle**.



CHAPTER 12

PKI Split VRF in Trustpoint

The PKI Split VRF in Trustpoint feature allows you to configure a VPN Routing and Forwarding (VRF) for certificate enrollment and revocation.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About PKI Split VRF in Trustpoint

Overview of PKI Split VRF in Trustpoint

The PKI Split VRF in Trustpoint feature allows you to configure VPN Routing and Forwarding (VRF) for certificate enrollment and for certificate revocation list (CRL) checking. The VRF is configured in the enrollment profile using the **enrollment url** command under the **crypto pki profile enrollment** command to attach the enrollment profile to a trustpoint. You can configure the same VRF for enrollment and CRL or configure different VRFs. Based on the configuration (enrollment or revocation), the corresponding VRF is selected and Simple Certificate Enrollment Protocol (SCEP) request is sent via the respective VRF.

To configure enrollment and CRL via different routing paths, you must configure the enrollment url command using the **crypto pki profile enrollment** command. This configured VRF acts as an enrollment VRF and the enrollment request goes via that VRF. However, the CRL uses the global VRF configured in the trustpoint using the

If no VRF is configured in the **enrollment url** command, the enrollment takes global enrollment that is configured in the **crypto pki trustpoint** command.

How to Configure PKI Split VRF in Trustpoint

Configuring the Split VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki profile enrollment** *label*
4. **enrollment url** *url* [**vrf** *vrf-name*]
5. **exit**
6. **show crypto pki profile**
7. **show crypto pki trustpoint**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki profile enrollment** *label* **Example:** `Device(config)# crypto pki profile enrollment pki_profile`	Defines an enrollment profile and enters ca-profile-enroll configuration mode. • *label* —Name for the enrollment profile; the enrollment profile name must match the name specified in the **enrollment profile** command.
Step 4	**enrollment url** *url* [**vrf** *vrf-name*] **Example:** `Device(ca-profile-enroll)# enrollment url http://entrust:81/cda-cgi/clientcgi.exe vrf vrf1`	Specifies the URL and the VPN Routing and Forwarding (VRF) of the CA server to which to send certificate enrollment requests via HTTP or TFTP.
Step 5	**exit** **Example:** `Device(ca-profile-enroll)# exit`	Exits ca-profile-enroll configuration mode. • Enter this command a second time to exit global configuration mode.
Step 6	**show crypto pki profile** **Example:**	(Optional) Displays information about PKI profile.

	Command or Action	Purpose
	`Device# show crypto pki profile`	
Step 7	**show crypto pki trustpoint** **Example:** `Device# show crypto pki trustpoint`	(Optional) Displays information about PKI trustpoints.

Configuration Examples for PKI Split VRF in Trustpoint

Example: Configuring the PKI Split VRF in Trustpoint

Enrollment and Certificate Revocation List Via Same VRF

The following example shows how to configure the enrollment and certificate revocation list (CRL) via the same VRF:

```
crypto pki trustpoint trustpoint1
 enrollment url http://10.10.10.10:80
 vrf vrf1
 revocation-check crl
```

Enrollment and Certificate Revocation List Via Different VRF

The following example shows how to configure the enrollment and certificate revocation list (CRL) via different VRF:

```
crypto pki profile enrollment pki_profile
 enrollment url http://10.10.10.10:80 vrf vrf2

crypto pki trustpoint trustpoint1
 enrollment profile pki_profile
 vrf vrf1
 revocation-check crl
```

Additional References for PKI Split VRF in Trustpoint

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases

Related Topic	Document Title
Security commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z
Recommended cryptographic algorithms	Next Generation Encryption

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for PKI Split VRF in Trustpoint

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 16: Feature Information for PKI Split VRF in Trustpoint

Feature Name	Releases	Feature Information
PKI Split VRF in Trustpoint		The PKI Split VRF in Trustpoint feature allows you to configure a VPN Routing and Forwarding (VRF) for certificate enrollment and revocation. The following commands were introduced or modified: **enrollment url (ca-profile-enroll).**

CHAPTER 13

EST Client Support

The EST Client Support feature allows you to enable EST (Enrolment Over Secure Transport) for all trustpoints while using SSL or TLS to secure transport.

Feature Information for EST Client Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 17: Feature Information for EST Client Support

Feature Name	Releases	Feature Information
EST Client Support		The EST Client Support feature allows you to enable EST (Enrolment Over Secure Transport) for all trustpoints while using SSL or TLS to secure transport. The following command was introduced: **method-est**

Information About EST Client Support

Overview of EST Client Support

The EST Client Support feature allows you to use Enrollment over Secure Transport (EST) as a certificate management protocol for provisioning certificates. With the existing SCEP enrollment integrated within the

PKI component, the addition of EST will introduce a new component that will use SSL or TLS to secure the transport. PKI will store all certificates.

To enable EST support, the EST client is required to authenticate the server during TLS connection establishment. For this authentication, the TLS server may require the client's credentials.

Prerequisites for EST Client Support

- Enable the **ip http authentication fore-close** command.

Restrictions for EST Client Support

- The EST client supports only TLS 1.2
- The certificate Attribute request is not supported.
- CA-Certificate rollover is not supported.
- Certificate-less TLS authentication is not supported.

How to Configure EST Client Support

Configuring a Trustpoint to Use EST

Perform this task to configure a trustpoint to use EST (Enrolment Over Secure Transport) by enabling the user to use the enrollment profile.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki profile enrollment***label*
4. **method-est**
5. **enrollment url***url* [**vrf** *vrf name*]
6. **enrollment credential** *label*
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**crypto pki profile enrollment***label* **Example:** `Device(config)# crypto pki profile enrollment pki_profile`	Defines an enrollment profile and enters ca-profile-enroll configuration mode. • *label*—Name for the enrollment profile; the enrollment profile name must match the name specified in the **enrollment profile** command.
Step 4	**method-est** **Example:** `Device(ca-profile-enroll)# method-est`	Enables enrollment profile to select usage of EST.
Step 5	**enrollment url***url* [**vrf** *vrf name*] **Example:** `Device(ca-profile-enroll)# enrollment url http://entrust:81/cda-cgi/clientcgi.exe vrf vrf1`	Specifies that an enrollment profile is to be used for certificate enrollment. **Note** If the authentication URL is not specified, then the enrollment URL will be considered for authentication.
Step 6	**enrollment credential** *label* **Example:** `Device(ca-profile-enroll)# enrollment credential test_label`	Provides the trustpoint credentials currently available in the profile for TLS client authentication.
Step 7	**exit** **Example:** `Device(ca-profile-enroll)# exit`	Exits ca-profile-enroll configuration mode.

Verifying the EST Client Support Configaration

You can use the following show commands to verify EST Client Support configuration.

- **show crypto pki profile**
- **show crypto pki trustpoints estclient status**

Configuration Examples for EST Client Support

Configuring a Trustpoint to Use EST

The following example shows how to configure a trustpoint to use Enrollment over Secure Transport (EST):

```
crypto pki profile enrollment pki_profile
```

```
 method-est
 enrollment url http://www.example.com/BigCA/est/simpleenroll.dll
 enrollment credential test_label
```

Verifying EST Client Support

The following sample output from the **show crypto pki trustpoints estclient status** command verifies EST Client Support configuration.

```
Router# show crypto pki trustpoints estclient status
Trustpoint estclient:
  Issuing CA certificate configured:
    Subject Name:
     cn=estExampleCA
    Fingerprint MD5: B9D0403C 7D33F1AA F9957796 CA6E86AA
    Fingerprint SHA1: F3698C9C DCB2B5F2 A38EBCB4 1DBA6A90 9F877A5B
  Router Signature certificate configured:
    Subject Name:
     cn=estclientrouter
    Fingerprint MD5: B740849B 37016DB7 A6797CE4 D6140D27
    Fingerprint SHA1: F032B015 50BB5742 2619EFC6 F1F0B8B1 31D9906D
  State:
    Keys generated ............. Yes (Signature, non-exportable)
    Issuing CA authenticated ....... Yes
    Certificate request(s) ..... Yes
```

The following sample output from the **show crypto pki certificate estclient** command shows the status before re-enrollement and after re-enrollment.

```
BEFORE REENROLLMENT

Router# show crypto pki certificate estclient


Certificate
  Status: Available
  Certificate Serial Number (hex): 2603
  Certificate Usage: Signature
  Issuer:
    cn=estExampleCA
  Subject:
    Name: estclientrouter
    cn=estclientrouter
  CRL Distribution Points:
    http://example.com/crl.pem
  Validity Date:
    start date: 19:31:24 GMT Feb 8 2019
    end   date: 19:31:24 GMT Feb 8 2020
    renew date: 19:35:50 GMT Feb 8 2019
  Associated Trustpoints: estclient

CA Certificate
  Status: Available
  Certificate Serial Number (hex): 00ACFCD09D3182CBEB
  Certificate Usage: General Purpose
  Issuer:
    cn=estExampleCA
  Subject:
    cn=estExampleCA
  Validity Date:
    start date: 09:40:47 GMT Mar 28 2018
    end   date: 09:40:47 GMT Mar 28 2019
  Associated Trustpoints: estclient ROOT


AFTER REENROLLMENT

show crypto pki certificates estclient
Certificate
  Status: Available
  Certificate Serial Number (hex): 4B
  Certificate Usage: Signature
  Issuer:
    cn=estExampleCA
  Subject:
    Name: estclientrouter
    cn=estclientrouter
  CRL Distribution Points:
    http://example.com/crl.pem
  Validity Date:
    start date: 07:34:05 GMT Feb 9 2019
    end   date: 07:34:05 GMT Feb 9 2020
    renew date: 19:38:35 GMT Feb 8 2019
  Associated Trustpoints: estclient

CA Certificate
  Status: Available
  Certificate Serial Number (hex): 00E5EEC53E0FBD597D
  Certificate Usage: General Purpose
  Issuer:
    cn=estExampleCA
  Subject:
    cn=estExampleCA
  Validity Date:
    start date: 04:59:30 GMT Dec 20 2018
    end   date: 04:59:30 GMT Dec 20 2019
  Associated Trustpoints: estclient ROOT_SEC
```

Additional References for EST Client Support

Related Documents

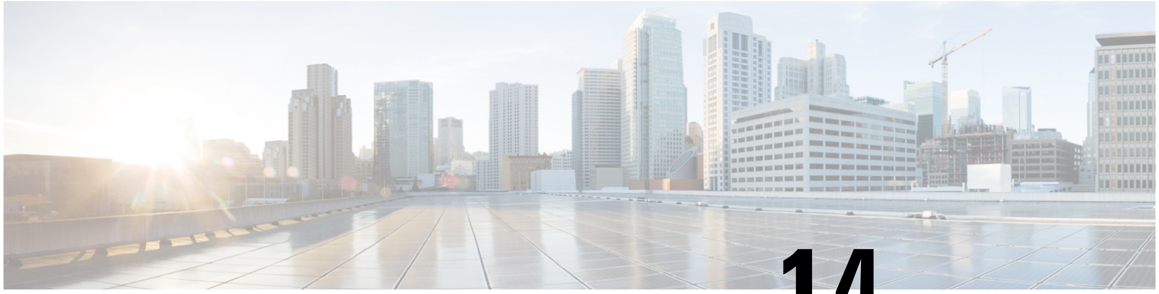
Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z

Standards and RFCs

Standard/RFC	Title
RFC 7030	*Enrollment over Secure Transport*
RFC 2818	*HTTP Over TLS*
RFC 6125	*Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)*
RFC 2510	*Internet X.509 Public Key Infrastructure Certificate Management Protocols*
RFC 4210	*Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 14

OCSP Response Stapling

The OCSP Response Stapling feature allows you to check the validity of a peer's user or device credentials contained in a digital certificate using Online Certificate Status Protocol (OCSP).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About OCSP Response Stapling

Overview of OCSP Response Stapling

Online Certificate Status Protocol (OCSP) is a method to check certificate revocation when a peer has to retrieve this revocation information and then validate it to check the certificate revocation status. In this method, the certification revocation status is limited by the peer's ability to reach an OCSP responder through the cloud or by the certificate sender's performance in retrieving the certificate revocation-information.

OCSP response stapling supports a new method to fetch the OCSP response for a device’s own certificates. This feature allows the device to obtain its own certificate revocation information by contacting the OCSP server and then sending this result along with its certificates directly to the peer. As a result, the peer does not require to contact the OCSP responder.

How to Configure OCSP Response Stapling

Configuring PKI Client to Request EKU Attribute

Perform this task to configure OCSP (Online Certificate Status Protocol) response stapling.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **ocsp url** *url*
5. **eku request** *attribute*
6. **match eku** *attribute*
7. **revocation-check** *method1* [*method2* [*method3*]]
8. **exit**
9. **exit**
10. **show cry pki counters**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **a.** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint msca`	Declares the trustpoint and a given name and enters ca-trustpoint configuration mode.

Step 4	**ocsp url** *url*
	Example:
	`Device(ca-trustpoint)# ocsp url http://ocsp-server`
	Example:
	`Device(ca-trustpoint)# ocsp url http://10.10.10.1:80`
	Example:
		The *url* argument specifies the URL of an OCSP server so that the trustpoint can check the certificate status. This URL overrides the URL of the OCSP server (if one exists) in the Authority Info Access (AIA) extension of the certificate. All certificates associated with a configured trustpoint are checked by the OCSP server. The URL can be a hostname, IPv4 address, or an IPv6 address.
		Note Make sure that the OCSP request url is configured with the **ocsp url** *url* command and not with an http-proxy server.

	Command or Action	Purpose
	`Device(ca-trustpoint)# ocsp url http://[2001DB8:1:1::2]:80`	
Step 5	**eku request** *attribute* **Example:** `Device(ca-trustpoint)# eku request ssh-client`	Requests to include specified eku *attribute* in the certificate. This request, when configured on the PKI client, will be sent to the CA server during enrollment. The *attribute* argument can be one of the following: • client-auth • code-signing • email-protection • ipsec-end-system • ipsec-tunnel • ipsec-user • ocsp-signing • server-auth • time-stamping • ssh-server • ssh-client
Step 6	**match eku** *attribute* **Example:** `Device(ca-trustpoint)# match eku client-auth`	Allows PKI to validate a peer certificate only if the specified attribute is present in the certificate else validation fails. The *attribute* argument can be one of the following: • client-auth • code-signing • email-protection • ipsec-end-system • ipsec-tunnel • ipsec-user • ocsp-signing • server-auth • time-stamping • ssh-server • ssh-client

	Command or Action	Purpose
Step 7	**revocation-check** *method1* [*method2* [*method3*]] **Example:** `Device(ca-trustpoint)# revocation-check ocsp none`	(Optional) Checks the revocation status of a certificate. • crl --Certificate checking is performed by a CRL. This is the default option. • none --Certificate checking is ignored. • ocsp --Certificate checking is performed by an OCSP server. If a second and third method are specified, each method will be used only if the previous method returns an error, such as a server being down.
Step 8	**exit** **Example:** `Device(ca-trustpoint)# exit`	Exits ca-trustpoint configuration mode and returns to global configuration mode.
Step 9	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 10	**show cry pki counters** **Example:** `Device# show cry pki counters`	(Optional) Displays the PKI counters of the device.

Configuring PKI Server to Include EKU Attributes

Perform this task to configure OCSP (Online Certificate Status Protocol) response stapling.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http server**
4. **crypto pki server** *cs-label*
5. **eku request** *attribute*
6. **exit**
7. **exit**
8. **show crypto pki counters**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. **a.** Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http server** **Example:** `Device(config)# ip http server`	Enables the HTTP server on your system.
Step 4	**crypto pki server** *cs-label* **Example:** `Device(config)# crypto pki server server-pki`	Defines a label for the certificate server and enters certificate server configuration mode. **Note** If you manually generated an RSA key pair, the *cs-label* argument must match the name of the key pair.
Step 5	**eku request** *attribute* **Example:** `Device(cs-server)# eku request ssh-server`	Requests to include specified eku *attribute* in the certificate. The *attribute* argument can be one of the following: • client-auth • code-signing • email-protection • ipsec-end-system • ipsec-tunnel • ipsec-user • ocsp-signing • server-auth • time-stamping • ssh-server • ssh-client
Step 6	**exit** **Example:** `Device(cs-server)# exit`	Exits cs-server configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 8	**show crypto pki counters** **Example:** `Device# show crypto pki counters`	(Optional) Displays the PKI counters of the device.

Example

The following is sample output from the **show crypto pki counters**.

```
Device# show crypto pki counters

PKI Sessions Started: 0
PKI Sessions Ended: 0
PKI Sessions Active: 0
Successful Validations: 0
Failed Validations: 0
Bypassed Validations: 0
Pending Validations: 0
CRLs checked: 0
CRL - fetch attempts: 0
CRL - failed attempts: 0
CRL - rejected busy fetching: 0
OCSP - fetch requests: 0
OCSP - received responses: 0
OCSP - failed attempts: 0
OCSP - staple requests: 0
AAA authorizations: 0
```

Additional References for OCSP Response Stapling

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference Commands A to C • Cisco IOS Security Command Reference Commands D to L • Cisco IOS Security Command Reference Commands M to R • Cisco IOS Security Command Reference Commands S to Z

Standards and RFCs

Standard/RFC	Title
RFC 2560	*X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*
RFC 4806	*Online Certificate Status Protocol (OCSP) Extensions to IKEv2*
RFC 5280	*Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*
RFC 6187	*X.509v3 Certificates for Secure Shell Authentication*
RFC 6066	*Transport Layer Security (TLS) Extensions: Extension Definitions*

MIBs

MIB	MIBs Link
	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for OCSP Response Stapling

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 18: Feature Information for OCSP Response Stapling

Feature Name	Releases	Feature Information
OCSP Response Stapling		This feature allows you to check the validity of a peer’s user or device credentials contained in a digital certificate using Online Certificate Status Protocol (OCSP).

CHAPTER 15

Configuring Route Processor Redundancy for PKI

Route Processor Redundancy provides an alternative to the High System Availability feature. HSA enables a system to reset and use a standby Route Switch Processor, if the active RSP fails. Using RPR, you can reduce unplanned downtime because RPR enables a quicker switchover between an active and standby RSP if the active RSP experiences a fatal error.

Route Processor Redundancy feature currently available on Cisco ASR platforms with dual RP support such as ASR 1006, ASR 1009. and ASR 1013.

Note Route Processor Redundancy supports trustpool import.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring Route Processor Redundancy

- You must use the same memory in both RSPs because the secondary RSP must be able to support the primary RSP during a failover.

Restrictions for Configuring Route Processor Redundancy

- Route Processor Redundancy feature only supports platforms with dual RP support.
- Route Processor Redundancy is supported only on routers that support dual RSPs.
- It is not recomended to configure RA (Registration Authority) as it is not validated.

How To Configure Route Processor Redundancy

Configuring Route Processor Redundancy SSO Mode

```
configure terminal
redundancy
 mode sso
 main-cpu
  standby console enable
exit
```

Verifying Route Processor Redundancy

```
show redundancy states
show crypto pki server
show crypto pki certificates tname
```

Route Processor Redundancy SSO Mode Confuguration Example

Example for server side configuration:

```
asr1k(config)#ip http server

asr1k(config)#crypto pki trustpoint ROOTCA

asr1k(ca-trustpoint)#hash sha512

asr1k(ca-trustpoint)#revocation-check none

asr1k(ca-trustpoint)#rsakeypair  ROOTCA 2048

asr1k(ca-trustpoint)#crypto pki server ROOTCA

asr1k(cs-server)#issuer-name CN=ROOTCA C=pki

asr1k(cs-server)#lifetime certificate 00 00 15

asr1k(cs-server)#lifetime ca-certificate 00 00 25

asr1k(cs-server)#lifetime crl 6

asr1k(cs-server)#serial-number 0x1
```

```
asr1k(cs-server)#auto-rollover 00 00 24

% The archive password is not configured. Rollover CA keys and certificates will not be
automatically archived.

asr1k(cs-server)#grant auto

asr1k(cs-server)#database url tftp://<ip>//

% Server database url was changed. You need to move the

% existing database to the new location.

asr1k(cs-server)#database url p12 tftp://<ip>//

asr1k(cs-server)#database level complete

asr1k(cs-server)#database archive pkcs12 password <pwd>

asr1k(cs-server)#end
```

Example for client side configuration:

```
crypto pki trustpoint client

 enrollment url http://<ip>:80

 usage ike

 subject-name CN=R1 C=pki

 revocation-check crl

 rsakeypair client 2048

 hash sha512
```

Route Processor Redundancy SSO Mode Verification Example

```
show redundancy states

       my state = 13 -ACTIVE

     peer state = 8  -STANDBY HOT

           Mode = Duplex

           Unit = Primary

        Unit ID = 48


Redundancy Mode (Operational) = sso

Redundancy Mode (Configured)  = sso

Redundancy State              = sso

     Maintenance Mode = Disabled
```

```
    Manual Swact = enabled
 Communications = Up

   client count = 132
 client_notification_TMR = 30000 milliseconds
           RF debug mask = 0x0
```

```
show crypto pki server
Certificate Server ROOTCA:
    Status: enabled
    State: enabled
    Server's configuration is locked  (enter "shut" to unlock it)
    Issuer name: CN=ROOTCA C=pki
    CA cert fingerprint: F2BF3707 D9F6F5F3 E0D111D8 A8486437
    Granting mode is: auto
    Last certificate issued serial number (hex): 2
    CA certificate expiration timer: 14:15:50 IST Mar 31 2019
    CRL NextUpdate timer: 14:15:50 IST Mar 31 2019
    Current primary storage dir: tftp://9.45.3.3//
    Current storage dir for .p12 files: tftp://9.45.3.3//
    Database Level: Complete - all issued certs written as <serialnum>.cer
    Auto-Rollover configured, overlap period 0 days
    Autorollover timer: 13:51:50 IST Mar 31 2019
    Redundancy configured.  This is active.
```

Note Server is enabled only on active RP and is in disabled state in standby mode.

```
show crypto pki certificates client
Certificate
  Status: Available
  Certificate Serial Number (hex): 03
  Certificate Usage: General Purpose
  Issuer:
    cn=ROOTCA C=pki
  Subject:
    Name: asr1k
    hostname=asr1k
    cn=R1 C=pki
  Validity Date:
    start date: 00:42:04 IST Mar 11 2019
    end   date: 01:02:04 IST Mar 11 2019
  Associated Trustpoints: client

CA Certificate
  Status: Available
  Certificate Serial Number (hex): 02
  Certificate Usage: Signature
  Issuer:
    cn=ROOTCA C=pki
  Subject:
    cn=ROOTCA C=pki
  Validity Date:
    start date: 00:40:34 IST Mar 11 2019
    end   date: 00:40:34 IST Mar 9 2020
  Associated Trustpoints: client
```