

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.

- Prerequisites for Configuring a Certificate Server, on page 2
- Restrictions for Configuring a Certificate Server, on page 2
- Information About Certificate Servers, on page 3
- How to Set Up and Deploy a Certificate Server, on page 10
- Configuration Examples for Using a Certificate Server, on page 38
- Where to Go Next, on page 49
- Additional References for Configuring and Managing a Certificate Server for PKI Deployment, on page 49
- Feature Information for Overview of Cisco TrustSec, on page 50

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 1: 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.

File Extension	File Type
.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 2: Certificate Server Publish File Types

File Extension	File Type
.crl	The CRL publish location.
.crt	The issued certificates publish location.
.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 3: 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**]	Generates the RSA key pair for the certificate server.

	Command or Action	Purpose
	[**modulus** *modulus-size*] [**storage** *devicename:*] [**on** *devicename:*] **Example:** `Device(config)# crypto key generate rsa label mycs exportable modulus 2048`	• 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.
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.

	Command or Action	Purpose
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:** `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	**no shutdown**	(Optional) Enables the certificate server.

	Command or Action	Purpose
	Example: `Device(cs-server)# no shutdown`	**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:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
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.
		• (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

	Command or Action	Purpose
		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 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.

	Command or Action	Purpose
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).
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).

	Command or Action	Purpose
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:** `Device(cs-server)# no shutdown`	Enables the certificate server. **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:**	Reenables the certificate server.

	Command or Action	Purpose
	`Device(cs-server)# no shutdown`	

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:** `Device# crypto pki server root-server grant 9`	Grants the pending RA certificate request. **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.

	Command or Action	Purpose
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. **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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
		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 Overview of Cisco TrustSec

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 Overview of Cisco TrustSec

Feature Name	Releases	Feature Information
IPv6 enablement - Inline Tagging	Cisco IOS XE Fuji 16.8.1	The support for IPv6 is introduced.