crypto pki authenticate

To authenticate the certification authority (CA) (by getting the certificate of the CA), use the crypto pki authenticate command in global configuration mode.

```
crypto pki authenticate name
```

**Syntax Description**

| name     | The name of the CA. This is the same name used when the CA was declared with the crypto ca identity command. |

**Defaults**

No default behavior or values.

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.3T</td>
<td>The crypto ca authenticate command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the crypto ca authenticate command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>12.4(24)T</td>
<td>Support for IPv6 Secure Neighbor Discovery (SeND) was added.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command is required when you initially configure CA support at your router.

This command authenticates the CA to your router by obtaining the self-signed certificate of the CA that contains the public key of the CA. Because the CA signs its own certificate, you should manually authenticate the public key of the CA by contacting the CA administrator when you enter this command.

If you are using Router Advertisements (RA) mode (using the enrollment command) when you issue the crypto pki authenticate command, then registration authority signing and encryption certificates will be returned from the CA and the CA certificate.

This command is not saved to the router configuration. However, the public keys embedded in the received CA (and RA) certificates are saved to the configuration as part of the Rivest, Shamir, and Adelman (RSA) public key record (called the “RSA public key chain”).

**Note**

If the CA does not respond by a timeout period after this command is issued, the terminal control will be returned so that it remains available. If this happens, you must reenter the command. Cisco IOS software will not recognize CA certificate expiration dates set for beyond the year 2049. If the validity period of the CA certificate is set to expire after the year 2049, the following error message will be displayed when authentication with the CA server is attempted:

```
error retrieving certificate :incomplete chain
```
If you receive an error message similar to this one, check the expiration date of your CA certificate. If the expiration date of your CA certificate is set after the year 2049, you must reduce the expiration date by a year or more.

**Examples**

In the following example, the router requests the certificate of the CA. The CA sends its certificate and the router prompts the administrator to verify the certificate of the CA by checking the CA certificate’s fingerprint. The CA administrator can also view the CA certificate’s fingerprint, so you should compare what the CA administrator sees to what the router displays on the screen. If the fingerprint on the router’s screen matches the fingerprint viewed by the CA administrator, you should accept the certificate as valid.

```
Router(config)# crypto pki authenticate myca
Certificate has the following attributes:
Fingerprint: 0123 4567 89AB CDEF 0123
Do you accept this certificate? [yes/no] y#
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>debug crypto pki transactions</td>
<td>Displays debug messages for the trace of interaction (message type) between the CA and the router.</td>
</tr>
<tr>
<td>enrollment</td>
<td>Specifies the enrollment parameters of your CA.</td>
</tr>
<tr>
<td>show crypto pki certificates</td>
<td>Displays information about your certificate, the certificate of the CA, and any RA certificates.</td>
</tr>
</tbody>
</table>
crypto pki benchmark

To start or stop benchmarking data for Public Key Infrastructure (PKI) performance monitoring and optimization, use the `crypto pki benchmark` command in privileged EXEC mode.

```
crypto pki benchmark {start limit [wrap] | stop}
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>start limit</code></td>
<td>Enables PKI benchmarking. The <code>limit</code> argument states the number of records from 0 to 9990 that can be stored for the benchmarking session. A limit of 0 indicates an unlimited number of records can be stored.</td>
</tr>
<tr>
<td><code>wrap</code></td>
<td>(Optional) Specifies a continuous flow of records. Once the maximum number of records is gathered, they are released and a new set of records is generated. If the <code>wrap</code> keyword is not specified, then benchmarking stops once the limit for the maximum number of records has been reached.</td>
</tr>
<tr>
<td><code>stop</code></td>
<td>Terminates PKI benchmarking data collection.</td>
</tr>
</tbody>
</table>

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.1(3)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Use the `crypto pki benchmark start` command to start the collection of PKI benchmarking performance monitoring and optimization data. Use the `crypto pki benchmark stop` command to stop the collection of the PKI benchmarking performance monitoring and optimization data.

Use the `show crypto pki benchmarks` command to view the collection data.

Use the `clear crypto pki benchmarks` command to clear the PKI benchmarking performance monitoring and optimization data and release all memory associated with this data.

The IOS PKI Performance Monitoring and Optimization feature enables you to collect the following types of PKI performance data:

- Time to validate entire certificate chain.
- Time to verify each certificate.
- Time to check revocation status for each certificate.
- Time to fetch certificate revocation list (CRL) database for each fetch location.
- Time to fetch Simple Certificate Enrollment Protocol (SCEP) method capabilities to retrieve the CRL.
- Time to process each CRL.
- Time to process the Online Certificate Status Protocol (OCSP) response. OCSP is a certificate revocation mechanism.
- Time to fetch Authentication, Authorization, and Accounting (AAA).
- CRL size.
- Validation result.
- Validation Bypass (pubkey cached).
- Method used to fetch a CRL.
- PKI session identifier.
- Crypto engine used (hardware, software, etoken).

**Examples**

The following example starts PKI benchmarking data and collects 20 records. Once 20 records are collected, they are released and a new set of 20 records is generated.

```
Router# crypto pki benchmark start 20 wrap
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear crypto pki benchmarks</td>
<td>Clears PKI benchmarking performance monitoring and optimization data and releases all memory associated with this data.</td>
</tr>
<tr>
<td>show crypto pki benchmarks</td>
<td>Displays benchmarking data for PKI performance monitoring and optimization that was collected.</td>
</tr>
</tbody>
</table>
crypto pki cert validate

To determine if a trustpoint has been successfully authenticated, a certificate has been requested and granted, and if the certificate is currently valid, use the `crypto pki cert validate` command in global configuration mode.

```
crypto pki cert validate trustpoint
```

**Syntax Description**

| `trustpoint` | The trustpoint to be validated. |

**Defaults**

No default behavior or values.

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(8)T</td>
<td>This command was introduced. Also, effective with Cisco IOS Release 12.3(8)T, this command replaced the <code>crypto ca cert validate</code> command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The `crypto pki cert validate` command validates the router's own certificate for a given trustpoint. Use this command as a sanity check after enrollment to verify that the trustpoint is properly authenticated, a certificate has been requested and granted for the trustpoint, and that the certificate is currently valid. A certificate is valid if it is signed by the trustpoint certification authority (CA), not expired, and so on.

**Examples**

The following examples show the possible output from the `crypto pki cert validate` command:

```
Router(config)# crypto pki cert validate ka
Validation Failed: trustpoint not found for ka

Router(config)# crypto pki cert validate ka
Validation Failed: can't get local certificate chain

Router(config)# crypto pki cert validate ka
Certificate chain has 2 certificates.
Certificate chain for ka is valid

Router(config)# crypto pki cert validate ka
```
Certificate chain has 2 certificates.
Validation Error: no certs on chain

Router(config)# crypto pki cert validate ka
Certificate chain has 2 certificates.
Validation Error: unspecified error

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the certification authority that the router should use.</td>
</tr>
<tr>
<td>show crypto pki trustpoints</td>
<td>Displays the trustpoints that are configured in the router.</td>
</tr>
</tbody>
</table>
crypto pki certificate chain

To enter the certificate chain configuration mode, use the `crypto pki certificate chain` command in global configuration mode.

```
crypto pki certificate chain name
```

**Syntax Description**

- **name**
  - Specifies the name of the certificate authority (CA). The name must match that which was declared for the CA using the `crypto pki trustpoint` command.

**Defaults**

No default behavior or values.

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.3 T</td>
<td>The <code>crypto ca certificate chain</code> command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca certificate chain</code> command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.4(2)T</td>
<td>The command output was modified to distinguish the current active certificate and the rollover certificate in the certificate chain.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command puts you into certificate chain configuration mode. When you are in certificate chain configuration mode, you can delete certificates using the `certificate` command.

You need to be in certificate chain configuration mode to delete certificates.

**Examples**

The following example deletes the router’s certificate. In this example, the router had a general-purpose RSA key pair with one corresponding certificate. The show command is used to determine the serial number of the certificate to be deleted.

```
Router# show crypto pki certificates

Certificate
  Subject Name
    Name: myrouter.example.com
    IP Address: 10.0.0.1
  Status: Available
  Certificate Serial Number: 0123456789ABCDEF0123456789ABCDEF
  Key Usage: General Purpose

CA Certificate
  Status: Available
  Certificate Serial Number: 3051DF7123BEE31B8341DFE4B3A338E5F
  Key Usage: Not Set
```
Router# configure terminal
Router(config)# crypto pki certificate chain myca
Router(config-cert-chain)# no certificate 0123456789ABCDEF0123456789ABCDEF
% Are you sure you want to remove the certificate [yes/no]? yes
% Be sure to ask the CA administrator to revoke this certificate.
Router(config-cert-chain)# exit

The following example shows a certificate chain with an active CA certificate and a shadow, or rollover, certificate:

Router# configure terminal
Router(config)# crypto pki certificate chain myca

certificate 06

certificate ca 01

certificate rollover 0B
! This is the peer's shadow PKI certificate.

certificate rollover ca 0A
! This is the CA shadow PKI certificate

This example shows how the certificate chain is rewritten when rollover actually happens:

Router# configure terminal
Router(config)# crypto pki certificate chain myca

certificate 0B

certificate ca 0A

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificate</td>
<td></td>
<td>Adds certificates manually.</td>
</tr>
</tbody>
</table>
crypto pki certificate map

To define certificate-based access control lists (ACLs), use the **crypto pki certificate map** command in ca-certificate-map configuration mode. To remove the certificate-based ACLs, use the **no** form of this command.

```
crypto pki certificate map label sequence-number
no crypto pki certificate map label sequence-number
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>A user-specified label that is referenced within the <strong>crypto pki trustpoint</strong> command.</td>
</tr>
<tr>
<td>sequence-number</td>
<td>A number that orders the ACLs with the same label. ACLs with the same label are processed from lowest to highest sequence number. When an ACL is matched, processing stops with a successful result.</td>
</tr>
</tbody>
</table>

**Defaults**

None

**Command Modes**

Ca-certificate-map configuration (ca-certificate-map)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.2(15)T</td>
<td>The <strong>crypto ca certificate map</strong> command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <strong>crypto ca certificate map</strong> command.</td>
</tr>
<tr>
<td>12.2(18)SXD</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXD.</td>
</tr>
<tr>
<td>12.4(9)T</td>
<td>The <strong>serial-number</strong> field name was introduced.</td>
</tr>
<tr>
<td>Cisco IOS XE Release 2.4</td>
<td>This command was implemented on the Cisco ASR 1000 series routers.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Issuing this command places the router in ca-certificate-map configuration mode where you can specify several certificate fields together with their matching criteria. The general form of these fields is as follows:

```
field-name match-criteria match-value
```

The **field-name** field in the above example is one of the certificate fields. Field names are similar to the names used in the ITU-T X.509 standard. The **field-name** is a special field that matches any subject name or related name field in the certificate, such as the **alt-subject-name**, **subject-name**, and **unstructured-subject-name** fields.

- **alt-subject-name**—Case-insensitive string.
- **expires-on**—Date field in the format dd mm yyyy hh:mm:ss or mmm dd yyyy hh:mm:ss.
- **issuer-name**—Case-insensitive string.
- **name**—Case-insensitive string.
• **serial-number**—Case-insensitive string.
• **subject-name**—Case-insensitive string.
• **unstructured-subject-name**—Case-insensitive string.
• **valid-start**—Date field in the format dd MM. yyy hh:mm:ss or mmm dd yyyy hh:mm:ss.

Note

The time portion is optional in both the **expires-on** date and **valid-start** field and defaults to 00:00:00 if not specified. The time is interpreted according to the time zone offset configured for the router. The string **utc** can be appended to the date and time when they are configured as Universal Time, Coordinated (UTC) rather than local time.

The **match-criteria** field in the example is one of the following logical operators:

• **eq**—equal (valid for name and date fields)
• **ne**—not equal (valid for name and date fields)
• **co**—contains (valid only for name fields)
• **nc**—does not contain (valid only for name fields)
• **lt**—less than (valid only for date fields)
• **ge**—greater than or equal to (valid only for date fields)

The **match-value** field is a case-insensitive string or a date.

Examples

The following example shows how to configure a certificate-based ACL that will allow any certificate issued by Company to an entity within the company.com domain. The label is Company, and the sequence is 10.

```plaintext
crypto pki certificate map Company 10
  issuer-name co Company
  unstructured-subject-name co company.com
```

The following example accepts any certificate issued by Company for an entity with DIAL or organizationUnit component ou=WAN. This certificate-based ACL consists of two separate ACLs tied together with the common label Group. Because the check for DIAL has a lower sequence number, it is performed first. Note that the string “DIAL” can occur anywhere in the subjectName field of the certificate, but the string WAN must be in the organizationUnit component.

```plaintext
crypto pki certificate map Group 10
  issuer-name co Company
  subject-name co DIAL

crypto pki certificate map Group 20
  issuer-name co Company
  subject-name co ou=WAN
```

Case is ignored in string comparisons; therefore, DIAL in the previous example will match dial, DIAL, Dial, and so on. Also note that the component identifiers (o=, ou=, cn=, and so on) are not required unless it is desirable that the string to be matched occurs in a specific component of the name. (Refer to the ITU-T security standards for more information about certificate fields and components such as ou=.)

If a component identifier is specified in the match string, the exact string, including the component identifier, must appear in the certificate. This requirement can present a problem if more than one component identifier is included in the match string. For example, “ou=WAN,o=Company” will not match a certificate with the string “ou=WAN,ou=Engineering,o=Company” because the “ou=Engineering” string separates the two desired component identifiers.
To match both “ou=WAN” and “o=Company” in a certificate while ignoring other component identifiers, you could use this certificate map:

```plaintext
crypto pki certificate map Group 10
  subject-name co ou=WAN
  subject-name co o=Company
```

Any space character proceeding or following the equal sign (=) character in component identifiers is ignored. Therefore “o=Company” in the proceeding example will match “o = Company,” “o =Company,” and so on.

The following example shows a CA map file used to certificate serial number session control:

```plaintext
crypto pki trustpoint CA1
  enrollment url http://CA1
  ip-address FastEthernet0/0
  crl query ldap://CA1_ldap
  revocation-check crl
  match certificate crl-map1

  crypto pki certificate map crl-map1 1
  serial-number ne 489d
```

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the CA that your router should use.</td>
</tr>
</tbody>
</table>
crypto pki certificate query (ca-trustpoint)

To specify that certificates should not be stored locally but retrieved from a certification authority (CA) trustpoint, use the `crypto pki certificate query` command in ca-trustpoint configuration mode. To cause certificates to be stored locally per trustpoint, use the `no` form of this command.

```
crypto pki certificate query
```

```
no crypto pki certificate query
```

**Syntax Description**  
This command has no arguments or keywords.

**Defaults**  
CA trustpoints are stored locally in the router’s NVRAM.

**Command Modes**  
Ca-trustpoint configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.2(8)T</td>
<td>The <code>crypto ca certificate query (ca-trustpoint)</code> command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca certificate query (ca-trustpoint)</code> command.</td>
</tr>
<tr>
<td>12.2(18)SXD</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXD.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Normally, certain certificates are stored locally in the router’s NVRAM, and each certificate uses a moderate amount of memory. To save NVRAM space, you can use this command to put the router into query mode, preventing certificates from being stored locally; instead, they are retrieved from a specified CA trustpoint when needed. This will save NVRAM space but could result in a slight performance impact.

The `crypto pki certificate query` command is a subcommand for each trustpoint; thus, this command can be disabled on a per-trustpoint basis.

Before you can configure this command, you must enable the `crypto pki trustpoint` command, which puts you in ca-trustpoint configuration mode.

**Note**

This command deprecates the `crypto ca certificate query` command in global configuration mode. Although you can still enter the global configuration command, the configuration mode and command will be written back as ca-trustpoint.
Examples

The following example shows how to prevent certificates and certificate revocation lists (CRLs) from being stored locally on the router; instead, they are retrieved from the “ka” trustpoint when needed.

crypto pki trustpoint ka
.
.
crypto pki certificate query

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the CA that your router should use.</td>
</tr>
</tbody>
</table>
crypto pki certificate storage

To specify the local storage location for public key infrastructure (PKI) credentials, use the **crypto pki certificate storage** command in global configuration mode. To restore the default behavior, that is to store PKI credentials to NVRAM, use the **no** form of this command.

```
crypto pki certificate storage location-name
```

```
no crypto pki certificate storage
```

**Syntax Description**

<table>
<thead>
<tr>
<th><strong>location-name</strong></th>
<th>Name of the local storage device.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Default is NVRAM.</td>
</tr>
</tbody>
</table>

**Defaults**

NVRAM is the default local storage location if this command is not issued.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th><strong>Release</strong></th>
<th><strong>Modification</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(2)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>12.2(33)SXH</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SXH.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

All Cisco platforms support NVRAM and flash local storage. Depending on your platform, you may have other supported local storage options including bootflash, slot, disk, USB flash, or USB token. During runtime, you can specify what active local storage device you would like to use to store PKI credentials. You must have the following system requirements before you can specify PKI credentials local storage location:

- A Cisco IOS Release 12.4(2)T PKI-enabled image or a later image
- A platform that supports storing PKI credentials as separate files
- A configuration that contains at least one certificate
- An accessible local file system

When using a local storage device to store PKI data, the following restrictions are applicable:

- Only local file systems may be used. An error message will be displayed if a remote file system is selected, and the command will not take effect.
- A subdirectory may be specified if supported by the local file system. NVRAM does not support subdirectories.
- Settings will take effect only when the running configuration is saved to the startup configuration.

If the keys are generated on the etoken, then the default storage location for the certificates is the etoken location.
for the device certificates. The CA certificates are stored in NVRAM. This allows for the credentials (keys and certificates) to be stored together on the removable media by default.

Examples

The following configuration example shows how to store certificates to the certs subdirectory. The certs subdirectory does not exist and is automatically created.

Router# dir nvram:

```
114 -rw- 4687 <no date> startup-config
115 ---- 5545 <no date> private-config
116 -rw- 4687 <no date> underlying-config
1 ---- 34 <no date> persistent-data
3 -rw- 707 <no date> ioscaroot#7401CA.cer
9 -rw- 863 <no date> msca-root#826E.cer
10 -rw- 759 <no date> msca-root#1BA8CA.cer
11 -rw- 863 <no date> msca-root#75B8.cer
24 -rw- 1149 <no date> storagename#6500CA.cer
26 -rw- 863 <no date> msca-root#83EE.cer
```

129016 bytes total (92108 bytes free)

Router# config terminal

```
Enter configuration commands, one per line. End with CNTL/Z.
```

Router(config)# crypto pki certificate storage disk0:/certs

```
Requested directory does not exist -- created
Certificates will be stored in disk0:/certs/
```

Router(config)# end

Router# write

```
*May 27 02:09:00:%SYS-5-CONFIG_I:Configured from console by consolemem
Building configuration...
[OK]
```

Router# directory disk0:/certs

```
Directory of disk0:/certs/

14 -rw- 707 May 27 2005 02:09:02 +00:00 ioscaroot#7401CA.cer
15 -rw- 863 May 27 2005 02:09:02 +00:00 msca-root#826E.cer
16 -rw- 759 May 27 2005 02:09:02 +00:00 msca-root#1BA8CA.cer
17 -rw- 863 May 27 2005 02:09:02 +00:00 msca-root#75B8.cer
18 -rw- 1149 May 27 2005 02:09:02 +00:00 storagename#6500CA.cer
19 -rw- 863 May 27 2005 02:09:02 +00:00 msca-root#83EE.cer
```

47894528 bytes total (20934656 bytes free)

! The certificate files are now on disk0/certs:

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>show crypto pki certificates storage</td>
<td>Displays the current PKI certificate storage location.</td>
<td></td>
</tr>
</tbody>
</table>
crypto pki crl cache

To set the maximum amount of volatile memory used to cache certificate revocation lists (CRLs), use the `crypto pki crl cache` command in privileged EXEC mode. To restore the default value, use the `no` form of this command.

```plaintext
crypto pki crl cache cache-size

no crypto pki crl cache cache-size
```

**Syntax Description**

- `cache-size` The maximum CRL cache size in kilobytes.
  - The default value is 512 kilobytes.
  - The value specified must be an integer. Specifying a cache size of zero disables CRL caching.

**Command Default**

The default CRL cache size is set to 512 kilobytes.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(20)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>Cisco IOS XE</td>
<td>This command was implemented on the Cisco ASR 1000 series routers.</td>
</tr>
<tr>
<td>Release 2.4</td>
<td></td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The CRL cache is a global cache that holds all CRLs downloaded by the router regardless of the trustpoint configuration. The impact on router memory depends upon the CRL cache size configured by the administrator. Configuring the CRL cache size allows the amount of memory used for the CRL cache to be reduced (for instance, if low memory conditions exist) or to be increased for better performance (for instance, when a large number of CRLs are being processed).

If the `crypto pki crl cache` command is issued, regardless of the CRL cache size value set, the CRL cache size will be included in the configuration. Issuing the `no crypto pki crl cache` command will remove the CRL cache size from the configuration.

When a CRL is stored in the CRL cache, it is condensed at least one-fifth of its original size. Therefore, more CRLs can be stored in the CRL cache than would be expected based on the CRL size before being cached.

**Note**

To configure CRL caching for a given trustpoint, you may issue either the `crl-cache none` or `crl cache delete-after` command. To disable caching of CRLs for a given trustpoint, use the `crl-cache none` command. To set a maximum age for CRLs in the cache for a given trustpoint, use the `crl cache delete-after` command.
Examples

The following example sets the maximum CRL cache size to 2048 kilobytes and then shows sample output of the `show crypto pki crls` command:

Router# crypto pki crl cache 2048
Router# show crypto pki crls

CRL Issuer Name:
  cn=ioscs,l=Anytown,c=US
LastUpdate: 02:53:41 GMT Mar 6 2007
Retrieved from CRL Distribution Point:
  ** CDP Not Published - Retrieved via SCEP
CRL DER is 475 bytes
CRL is stored in parsed CRL cache
Parsed CRL cache current size is 1705 bytes
Parsed CRL cache maximum size is 2048 bytes

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crl cache delete-after</td>
<td>Deletes a CRL from the cache after the specified number of minutes.</td>
</tr>
<tr>
<td>crl cache none</td>
<td>Disables caching of all CRLs.</td>
</tr>
<tr>
<td>crypto pki crl request</td>
<td>Requests that a new CRL be obtained immediately from the CA.</td>
</tr>
<tr>
<td>show crypto pki crls</td>
<td>Displays the current CRL on the router.</td>
</tr>
</tbody>
</table>
crypto pki crl request

To request that a new certificate revocation list (CRL) be obtained immediately from the certification authority, use the crypto pki crl request command in global configuration mode.

```
crypto pki crl request name
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the CA. This is the same name used when the CA was declared with the crypto pki trustpoint command.</td>
</tr>
</tbody>
</table>

| **Defaults** | Normally, the router requests a new CRL when it is verifying a certificate and there is no CRL cached. |

| **Command Modes** | Global configuration |

<table>
<thead>
<tr>
<th><strong>Command History</strong></th>
<th><strong>Release</strong></th>
<th><strong>Modification</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>11.3 T</td>
<td>The crypto ca crl request command was introduced.</td>
</tr>
<tr>
<td></td>
<td>12.3(7)T</td>
<td>This command replaced the crypto ca crl request command.</td>
</tr>
<tr>
<td></td>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td></td>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

| **Usage Guidelines** | A CRL lists all the certificates of the network device that have been revoked. Revoked certificates will not be honored by your router; therefore, any IPSec device with a revoked certificate cannot exchange IP Security traffic with your router. |
|                     | The first time your router receives a certificate from a peer, it will download a CRL from the CA. Your router then checks the CRL to make sure the certificate of the peer has not been revoked. (If the certificate appears on the CRL, it will not accept the certificate and will not authenticate the peer.) |
|                     | A CRL can be reused with subsequent certificates until the CRL expires. If your router receives the certificate of a peer after the applicable CRL has expired, it will download the new CRL. |
|                     | If your router has a CRL which has not yet expired, but you suspect that the contents of the CRL are out of date, use the crypto pki crl request command to request that the latest CRL be immediately downloaded to replace the old CRL. |
|                     | This command is not saved to the configuration. |

| **Note** | This command should be used only after the trustpoint is enrolled. |

<table>
<thead>
<tr>
<th><strong>Examples</strong></th>
<th>The following example immediately downloads the latest CRL to your router:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>crypto pki crl request</td>
</tr>
</tbody>
</table>
crypto pki enroll

To obtain the certificates for your router from the certificate authority (CA), use the `crypto pki enroll` command in global configuration mode. To delete a current enrollment request, use the `no` form of this command.

```
crypto pki enroll name

no crypto pki enroll name
```

### Syntax Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name</code></td>
<td>The name of the CA. Use the same name as when you declared the CA using the <code>crypto pki trustpoint</code> command.</td>
</tr>
</tbody>
</table>

### Defaults

No default behavior or values.

### Command Modes

Global configuration (config)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.3T</td>
<td>The <code>crypto ca enroll</code> command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca enroll</code> command.</td>
</tr>
<tr>
<td>12.3(14)T</td>
<td>The command was modified to include self-signed certificate information.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>12.2SX</td>
<td>This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.</td>
</tr>
<tr>
<td>12.4(24)T</td>
<td>Support for IPv6 Secure Neighbor Discovery (SeND) was added.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

This command requests certificates from the CA for all of your router’s Rivest, Shamir, and Adelmen (RSA) key pairs. This task is also known as enrolling with the CA. (Technically, enrolling and obtaining certificates are two separate events, but they both occur when this command is issued.)

Your router needs a signed certificate from the CA for each RSA key pairs of your router; if you previously generated general-purpose keys, this command obtains the one certificate corresponding to the one general-purpose RSA key pair. If you previously generated special-usage keys, this command obtains two certificates corresponding to each of the special-usage RSA key pairs.

If you already have a certificate for your keys you are prompted to remove the existing certificate first. (You can remove existing certificates with the `no certificate` command.)

The `crypto pki enroll` command is not saved in the router configuration.

### Note

If your router reboots after you issue the `crypto pki enroll` command but before you receive the certificates, you must reissue the command.
If you are using a Secure Shell (SSH) service, you should set up specific RSA key pairs (different private keys) for the trustpoint and the SSH service. (If the Public Key Infrastructure [PKI] and the SSH infrastructure share the same default RSA key pair, a temporary disruption of SSH service could occur. The RSA key pair could become invalid or change because of the CA system, in which case you would not be able to log in using SSH. You could receive the following error message: “key changed, possible security problem.”)

Responding to Prompts

When you issue the `crypto pki enroll` command, you are prompted a number of times.

You are prompted to create a challenge password. This password can be up to 80 characters in length. This password is necessary in the event that you ever need to revoke your router’s certificates. When you ask the CA administrator to revoke your certificate, you must supply this challenge password as a protection against fraudulent or mistaken revocation requests.

This password is not stored anywhere, so you need to remember this password.

If you lose the password, the CA administrator may still be able to revoke the router’s certificate but will require further manual authentication of the router administrator identity.

You are also prompted to indicate whether your router’s serial number should be included in the obtained certificate. The serial number is not used by IP Security (IPsec) or Internet Key Exchange, but may be used by the CA to either authenticate certificates or to later associate a certificate with a particular router. (Note that the serial number stored is the serial number of the internal board, not the one on the enclosure.) Ask your CA administrator if serial numbers should be included. If you are in doubt, include the serial number.

Normally, you would not include the IP address because the IP address binds the certificate more tightly to a specific entity. Also, if the router is moved, you would need to issue a new certificate. A router has multiple IP addresses, any of which might be used with IPsec.

If you indicate that the IP address should be included, you will then be prompted to specify the interface of the IP address. This interface should correspond to the interface that you apply your crypto map set to. If you apply crypto map sets to more than one interface, specify the interface that you name in the `crypto map local-address` command.

Examples

In the following example, a router with a general-purpose RSA key pair requests a certificate from the CA. When the router displays the certificate fingerprint, the administrator verifies this number by calling the CA administrator, which checks the number. The fingerprint is correct, so the router administrator accepts the certificate.

There can be a delay between when the router administrator sends the request and when the certificate is actually received by the router. The amount of delay depends on the CA method of operation.

```plaintext
Router(config)# crypto pki enroll myca
% 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: <mypassword>
```
Re-enter password: <mypassword>

% The subject name in the certificate will be: myrouter.example.com
% Include the router serial number in the subject name? [yes/no]: yes
% The serial number in the certificate will be: 03433678
% Include an IP address in the subject name [yes/no]? yes
Interface: ethernet0/0
Request certificate from CA [yes/no]? yes
% Certificate request sent to Certificate Authority
% The certificate request fingerprint will be displayed.
% The 'show crypto pki certificates' command will also show the fingerprint.

Some time later, the router receives the certificate from the CA and displays the following confirmation message:

Router(config)# Fingerprint: 01234567 89ABCDEF FEDCBA98 75543210

%CRYPTO-6-CERTRET: Certificate received from Certificate Authority

Router(config)#

If necessary, the router administrator can verify the displayed fingerprint with the CA administrator.

If there is a problem with the certificate request and the certificate is not granted, the following message is displayed on the console instead:

%CRYPTO-6-CERTREJ: Certificate enrollment request was rejected by Certificate Authority

The subject name in the certificate is automatically assigned to be the same as the RSA key pair’s name. In the example, the RSA key pair was named “myrouter.example.com.” (The router assigned this name.)

Requesting certificates for a router with special-usage keys would be the same as in the previous example, except that two certificates would have been returned by the CA. When the router received the two certificates, the router would have displayed the same confirmation message:

%CRYPTO-6-CERTRET: Certificate received from Certificate Authority

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto map local address</td>
<td>Specifies and names an identifying interface to be used by the crypto map for IPsec traffic.</td>
</tr>
<tr>
<td>debug crypto pki messages</td>
<td>Displays debug messages for the details of the interaction (message dump) between the CA and the router.</td>
</tr>
<tr>
<td>debug crypto pki transactions</td>
<td>Displays debug messages for the trace of interaction (message type) between the CA and the router.</td>
</tr>
<tr>
<td>show crypto pki certificates</td>
<td>Displays information about your certificate, the certificate of the CA, and any RA certificates.</td>
</tr>
</tbody>
</table>
crypto pki export pem

To export certificates and Rivest, Shamir, and Adelman (RSA) keys that are associated with a trustpoint in a privacy-enhanced mail (PEM)-formatted file, use the `crypto pki export pem` command in global configuration mode.

```
crypto pki export trustpoint pem {terminal | url url} {3des | des} passphrase [rollover]
```

### Syntax Description

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>trustpoint</code></td>
<td>Name of the trustpoint that the associated certificate and RSA key pair will export. The <code>trustpoint</code> argument must match the name that was specified via the <code>crypto pki trustpoint</code> command.</td>
</tr>
<tr>
<td><code>terminal</code></td>
<td>Certificate and RSA key pair that will be displayed in PEM format on the console terminal.</td>
</tr>
<tr>
<td><code>url</code></td>
<td>URL of the file system where your router should export the certificate and RSA key pairs.</td>
</tr>
<tr>
<td><code>3des</code></td>
<td>Export the trustpoint using the Triple Data Encryption Standard (3DES) encryption algorithm.</td>
</tr>
<tr>
<td><code>des</code></td>
<td>Export the trustpoint using the DES encryption algorithm.</td>
</tr>
<tr>
<td><code>passphrase</code></td>
<td>Passphrase that is used to encrypt the PEM file for import. <strong>Note</strong> The passphrase can be any phrase that is at least eight characters in length; it can include spaces and punctuation, excluding the question mark (?), which has special meaning to the Cisco IOS parser.</td>
</tr>
<tr>
<td><code>rollover</code></td>
<td>(Optional) Export certificate authority (CA) shadow, or rollover, certificate.</td>
</tr>
</tbody>
</table>

### Defaults

No default behavior or values

### Command Modes

Global configuration

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>The <code>crypto ca export pem</code> command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca export pem</code> command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.4(2)T</td>
<td>The <code>rollover</code> keyword was added.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

The `crypto pki export pem` command allows you to export certificate and RSA key pairs in PEM-formatted files. The PEM files can then be imported back into the Cisco IOS router (via the `crypto pki import pem` command) or other public key infrastructure (PKI) applications.
Examples

The following example shows how to generate and export the RSA key pair “aaa” and certificates of the router in PEM files that are associated with the trustpoint “mycs”:

Router(config)# crypto key generate rsa general-keys label aaa exportable
The name for the keys will be:aaa
Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose Keys. Choosing a key modulus greater than 512 may take a few minutes.

How many bits in the modulus [512]:
% Generating 512 bit RSA keys ...[OK]

Router(config)# crypto pki trustpoint mycs
Router(config)# enrollment url http://mycs
Router(config)# rsakeypair aaa
Router(config)# exit
Router(config)# crypto pki authenticate mycs
Certificate has the following attributes:
Fingerprint:C21514AC 12815946 09F635ED FBB6CF31
% Do you accept this certificate? [yes/no]:y
Trustpoint CA certificate accepted.

Router(config)# crypto pki enroll mycs
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this password to the CA Administrator in order to revoke your certificate.
For security reasons your password will not be saved in the configuration.
Please make a note of it.

Password:
Re-enter password:

% The fully-qualified domain name in the certificate will be:Router
% The subject name in the certificate will be:bizarro.cisco.com
% Include the router serial number in the subject name? [yes/no]:n
% Include an IP address in the subject name? [no]:n
Request certificate from CA? [yes/no]:y
% Certificate request sent to Certificate Authority
% The certificate request fingerprint will be displayed.
% The 'show crypto ca certificate' command will also show the fingerprint.

Router(config)# Fingerprint: 8DA777BC 08477073 A5BE2403 812DD157
%CRYPTO-6-CERTRET:Certificate received from Certificate Authority
00:29:11:

Router(config)# crypto pki export aaa pem terminal 3des cisco123
% CA certificate:
-----BEGIN CERTIFICATE-----
MIICAzCCAgIBAwIBAgIBATANBgkqhkiG9w0BAQUFADBOMQswCQYDVQQGEwJVUzES
waDeNOSI3WlDa0AWq5DkVhB066xq1h4h10TgJ9XJOCttj7HnWHK1LMcMVg
-----END CERTIFICATE-----
% Key name:aaa
Usage:General Purpose Key
-----BEGIN RSA PRIVATE KEY-----
Proc-Type:4, ENCRYPTED
DEK-Info:DES-EDE3-CBC, ED6B210B626BC81A

Urguv0jn\wGowWVqUQ2XR5nbz5YH12vGLumP/H/IxIaJuNjRVj\bAAUpGk7VnPCT87
-----END CERTIFICATE-----

kLCoJxZEv7JHc72gMku9uUlzLShFH5s1zAtoC0czfU4=
-----END RSA PRIVATE KEY-----

% Certificate:
-----BEGIN CERTIFICATE-----
MIICCgAwIBAgICIQUwDQYJKoZIhvcNAQEFBQAwTjELMAkGA1UEBhMCVVMx
<snip>
6xlBaIsuMxnHmr89KkKkYlU6
-----END CERTIFICATE-----

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki import pem</td>
<td>Imports certificates and RSA keys to a trustpoint from PEM-formatted files.</td>
</tr>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the CA that your router should use.</td>
</tr>
<tr>
<td>enrollment</td>
<td>Specifies the enrollment parameters of a CA.</td>
</tr>
</tbody>
</table>
crypto pki export pkcs12

To export Rivest, Shamir, and Adelman (RSA) keys within a PKCS12 file at a specified location, use the crypto pki export pkcs12 command in global configuration mode.

```
crypto pki export trustpointname pkcs12 destination url passphrase
```

**Syntax Description**

- **trustpointname**: Name of the trustpoint who issues the certificate that a user is going to export. When you export the PKCS12 file, the trustpoint name is the RSA key name.
- **destination url**: Location of the PKCS12 file to which a user wants to import the RSA key pair.
- **passphrase**: Passphrase that is used to encrypt the PKCS12 file for export.

**Defaults**

No default behavior or values

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.2(15)T</td>
<td>The crypto ca export pkcs12 command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the crypto ca export pkcs12 command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

The crypto pki export pkcs12 command creates a PKCS12 file that contains an RSA key pair. The PKCS12 file, along with a certificate authority (CA), is exported to the location that you specify with the destination URL. If you decide not to import the file to another router, you must delete the file.

**Security Measures**

Keep the PKCS12 file stored in a secure place with restricted access.

An RSA key pair is more secure than a passphrase because the private key in the key pair is not known by multiple parties. When you export an RSA key pair to a PKCS#12 file, the RSA key pair now is only as secure as the passphrase.

To create a good passphrase, be sure to include numbers, as well as both lowercase and uppercase letters. Avoid publicizing the passphrase by mentioning it in e-mail or cell phone communications because the information could be accessed by an unauthorized user.

**Examples**

The following example exports an RSA key pair with a trustpoint name “mytp” to a Flash file:

```
Router(config)# crypto pki export mytp pkcs12 flash:myexport mycompany
```
<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>crypto pki export pkcs12</strong></td>
<td></td>
</tr>
</tbody>
</table>

Imports RSA keys.

**crypto pki import pkcs12** Imports RSA keys.
crypto pki import

To import a certificate manually via TFTP or as a cut-and-paste at the terminal, use the crypto pki import command in global configuration mode.

```
crypto pki import name certificate
```

**Syntax Description**

- `name certificate` Name of the certification authority (CA). This name is the same name used when the CA was declared with the `crypto pki trustpoint` command.

**Defaults**

No default behavior or values

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.2(13)T</td>
<td>The <code>crypto ca import</code> command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca import</code> command.</td>
</tr>
<tr>
<td>12.2(18)SXD</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXD.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>12.4(24)T</td>
<td>Support for IPv6 Secure Neighbor Discovery (SeND) was added.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

You must enter the `crypto pki import` command twice if usage keys (signature and encryption keys) are used. The first time the command is entered, one of the certificates is pasted into the router; the second time the command is entered, the other certificate is pasted into the router. (It does not matter which certificate is pasted first.)

**Examples**

The following example shows how to import a certificate via cut-and-paste. In this example, the CA trustpoint is “MS.”

```
crypto pki trustpoint MS
enroll terminal
    crypto pki authenticate MS
!crypto pki enroll MS
crypto pki import MS certificate
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>crypto pki trustpoint</code></td>
<td>Declares the CA that your router should use.</td>
</tr>
<tr>
<td><code>enrollment</code></td>
<td>Specifies the enrollment parameters of your CA.</td>
</tr>
<tr>
<td><code>enrollment terminal</code></td>
<td>Specifies manual cut-and-paste certificate enrollment.</td>
</tr>
</tbody>
</table>
crypto pki import pem

to import certificates and Rivest, Shamir, and Adelman (RSA) keys to a trustpoint from privacy-enhanced mail (PEM)-formatted files, use the crypto pki import pem command in global configuration mode.

crypto pki import trustpoint pem [usage-keys] {terminal | url url} [exportable] passphrase

Syntax Description

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trustpoint</td>
<td>Name of the trustpoint that is associated with the imported certificates and RSA key pairs. The trustpoint argument must match the name that was specified via the crypto pki trustpoint command.</td>
</tr>
<tr>
<td>usage-keys</td>
<td>(Optional) Specifies that two RSA special usage key pairs will be imported (that is, one encryption pair and one signature pair), instead of one general-purpose key pair.</td>
</tr>
<tr>
<td>terminal</td>
<td>Certificates and RSA key pairs will be manually imported from the console terminal.</td>
</tr>
<tr>
<td>url url</td>
<td>URL of the file system where your router should import the certificates and RSA key pairs.</td>
</tr>
<tr>
<td>exportable</td>
<td>(Optional) Specifies that the imported RSA key pair can be exported again to another Cisco device such as a router.</td>
</tr>
<tr>
<td>passphrase</td>
<td>Passphrase that is used to encrypt the PEM file for import.</td>
</tr>
</tbody>
</table>

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

Defaults

No default behavior or values

Command Modes

Global configuration

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>The crypto ca import pem command was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the crypto ca import pem command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

Usage Guidelines

The crypto pki import pem command allows you import certificates and RSA key pairs in PEM-formatted files. The files can be previously exported from another router or generated from other public key infrastructure (PKI) applications.
Examples

The following example shows how to import PEM files to trustpoint “ggg” via TFTP:

Router(config)# crypto pki import ggg pem url tftp://10.1.1.2/johndoe/msca cisco1234

% Importing CA certificate...
Address or name of remote host [10.1.1.2]?
Destination filename [johndoe/msca.ca]?
Reading file from tftp://10.1.1.2/johndoe/msca.ca
Loading johndoe/msca.ca from 10.1.1.2 (via Ethernet0):
[OK - 1082 bytes]

% Importing private key PEM file...
Address or name of remote host [10.1.1.2]?
Destination filename [johndoe/msca.prv]?
Reading file from tftp://10.1.1.2/johndoe/msca.prv
Loading johndoe/msca.prv from 10.1.1.2 (via Ethernet0):
[OK - 573 bytes]

% Importing certificate PEM file...
Address or name of remote host [10.1.1.2]?
Destination filename [johndoe/msca.crt]?
Reading file from tftp://10.1.1.2/johndoe/msca.crt
Loading johndoe/msca.crt from 10.1.1.2 (via Ethernet0):
[OK - 1289 bytes]
% PEM files import succeeded.
Router(config)#

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki export pem</td>
<td>Exports certificates and RSA keys that are associated with a trustpoint in a PEM-formatted file.</td>
</tr>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the CA that your router should use.</td>
</tr>
<tr>
<td>enrollment</td>
<td>Specifies the enrollment parameters of a CA.</td>
</tr>
</tbody>
</table>
crypto pki import pkcs12

To import Rivest, Shamir, and Adelman (RSA) keys, use the `crypto pki import pkcs12` command in global configuration mode.

```
crypto pki import trustpointname pkcs12 source url passphrase
```

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>trustpointname</code></td>
<td>Name of the trustpoint who issues the certificate that a user is going to export or import. When importing, the trustpoint name will become the RSA key name.</td>
</tr>
<tr>
<td><code>source url</code></td>
<td>The location of the PKCS12 file to which a user wants to export the RSA key pair.</td>
</tr>
<tr>
<td><code>passphrase</code></td>
<td>Passphrase that must be entered to undo encryption when the RSA keys are imported.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Defaults</th>
<th>No default behavior or values</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Command Modes</th>
<th>Global configuration</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Command History</th>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12.2(15)T</td>
<td>The <code>crypto ca import pkcs12</code> command was introduced.</td>
</tr>
<tr>
<td></td>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca import pkcs12</code> command.</td>
</tr>
<tr>
<td></td>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td></td>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

| Usage Guidelines   | When you enter the `crypto pki import pkcs12` command, a key pair and a trustpoint are generated. If you then decide you want to remove the key pair and trustpoint that were generated, enter the `crypto key zeroize rsa` command to zeroize the key pair and enter the `no crypto pki trustpoint` command to remove the trustpoint. |

| Note               | After you import RSA keys to a target router, you cannot export those keys from the target router to another router. |

<table>
<thead>
<tr>
<th>Examples</th>
<th>In the following example, an RSA key pair that has been associated with the trustpoint “forward” is to be imported:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>Router(config)# crypto pki import forward pkcs12 flash:myexport mycompany</code></td>
</tr>
</tbody>
</table>
### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki export pkcs12</td>
<td>Exports RSA keys.</td>
</tr>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the CA that your router should use.</td>
</tr>
<tr>
<td>crypto key zeroize rsa</td>
<td>Deletes all RSA keys from your router.</td>
</tr>
</tbody>
</table>
crypto pki profile enrollment

To define an enrollment profile, use the **crypto pki profile enrollment** command in global configuration mode. To delete all information associated with this enrollment profile, use the **no** form of this command.

```
crypto pki profile enrollment label
no crypto pki profile enrollment label
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>label</code></td>
<td>Name for the enrollment profile; the enrollment profile name must match the name specified in the <strong>enrollment profile</strong> command.</td>
</tr>
</tbody>
</table>

**Defaults**

An enrollment profile does not exist.

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.2(13)ZH</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.3(4)T</td>
<td>This command was integrated into Cisco IOS Release 12.3(4)T.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <strong>crypto ca profile enrollment</strong> command.</td>
</tr>
<tr>
<td>12.2(18)SXE</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXE.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Before entering this command, you must specify a named enrollment profile using the **enrollment profile** in ca-trustpoint configuration mode.

After entering the **crypto pki profile enrollment** command, you can use any of the following commands to define the profile parameters:

- **authentication command**—Specifies the HTTP command that is sent to the certification authority (CA) for authentication.
- **authentication terminal**—Specifies manual cut-and-paste certificate authentication requests.
- **authentication url**—Specifies the URL of the CA server to which to send authentication requests.
- **enrollment command**—Specifies the HTTP command that is sent to the CA for enrollment.
- **enrollment terminal**—Specifies manual cut-and-paste certificate enrollment.
- **enrollment url**—Specifies the URL of the CA server to which to send enrollment requests.
- **parameter**—Specifies parameters for an enrollment profile. This command can be used only if the **authentication command** or the **enrollment command** is used.
The **authentication url**, **enrollment url**, **authentication terminal**, and **enrollment terminal** commands allow you to specify different methods for certificate authentication and enrollment, such as TFTP authentication and manual enrollment.

### Examples

The following example shows how to define the enrollment profile named “E” and associated profile parameters:

```plaintext
crypto pki trustpoint Entrust
enrollment profile E
serial

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

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki trustpoint</td>
<td>Declares the PKI trustpoint that your router should use.</td>
</tr>
<tr>
<td>enrollment profile</td>
<td>Specifies that an enrollment profile can be used for certificate authentication and enrollment.</td>
</tr>
</tbody>
</table>
**crypto pki server**

To enable a Cisco IOS certificate server and enter certificate server configuration mode or to immediately generate shadow certification authority (CA) credentials, use the `crypto pki server` command in global configuration mode. To disable a certificate server (which is the default functionality), use the `no` form of this command.

```
crypto pki server cs-label [rollover [cancel] [request pkcs10 terminal]] [redundancy] [show] [serial-number serial-number]
```

```
no crypto pki server cs-label
```

### Syntax Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
</table>
| `cs-label` | Name of the certificate server.  
**Note** The certificate server name should not exceed 13 characters. |
| `rollover` | (Optional) Immediately generates a shadow CA certificate.  
**Note** If the `auto-enroll` command has been issued with the `regenerate` keyword, shadow keys will also be generated.  
**Note** If the shadow certificate and keys are already present this command will fail. |
| `cancel` | (Optional) Deletes the exiting shadow CA certificate when used with the `rollover` keyword.  
Shadow keys will also be deleted if they exist. |
| `request pkcs10 terminal` | (Optional) Exports CA shadow certificate.  
Also exports shadow keys if they exist. |
| `redundancy` | (Optional) Synchronizes the server configuration with that of the standby CA. |
| `show` | (Optional) Displays the current configuration of the server being configured. |
| `serial-number serial-number` | (Optional) Specifies the next serial number to be issued, and updates the serial-number file. |

### Defaults

A certificate server is not enabled; the automatic CA certificate rollover process is not initiated.

### Command Modes

Global configuration (config)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(2)T</td>
<td>The <code>rollover</code>, <code>cancel</code>, and <code>request pkcs10 terminal</code> keywords were introduced to support automated CA certificate rollover functionality.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified. The <code>redundancy</code>, <code>show</code>, and <code>serial-number</code> keywords were added.</td>
</tr>
</tbody>
</table>
Usage Guidelines

A certificate server allows you to more easily deploy public key infrastructure (PKI) by defining default behavior, which limits user interface complexity. To define the functionality of the certificate server, you can use any of the following certificate server configuration mode commands:

- **database (certificate server)**—Requires a username or password to be issued when accessing a database storage location.
- **database level**—Controls what type of data is stored in the certificate enrollment database.
- **database url**—Specifies the location where all database entries for the certificate server will be written out.
- **grant automatic**—Specifies automatic certificate enrollment.

**Note**
This command can be used for testing and building simple networks; however, it is recommended that you do not issue this command if your network is generally accessible.

- **issuer-name**—Specifies the distinguished name (DN) as the CA issuer name for the certificate server.
- **lifetime (certificate server)**—Specifies the lifetime of the CA or a certificate.
- **lifetime crl**—Defines the lifetime of the certificate revocation list (CRL) that is used by the certificate server.
- **shutdown**—Allows a certificate server to be disabled without removing the configuration.

**Note**
All of these commands are optional; thus, any basic certificate server functionality that is not specified via the command-line interface (CLI) will use the default value.

Automated CA Certificate Rollover

CAs and their clients, have certificates with expiration dates that have to be reissued when the current certificate is about to expire. CAs also have key pairs used to sign client certificates. When the CA certificate is expiring it must generate a new certificate and possibly a new key pair. This process, called rollover, allows for continuous operation of the network while clients and the certificate server are switching from an expiring CA certificate to a new CA certificate.

Examples

The following example shows how to enable the certificate server “mycertserver”:

```
Router(config)# ip http server
Router(config)# crypto pki server mycertserver
Router(config-cs)# database url tftp://mytftp/johndoe/mycertserver
```

The following example shows how to disable the certificate server “mycertserver”:

```
Router(config)# no crypto pki server mycertserver
% This will stop the Certificate Server process and delete the server configuration
Are you sure you want to do this? [yes/no]: yes
% Do you also want to remove the associated trustpoint and signing certificate and key? [yes/no]: no
% Certificate Server Process stopped
```

The following example shows a shadow client certificate request from a terminal:
Router# 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-----
MIIBUTBuIBADASMRwDqYDVQQDBwdoZXNzb290MIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDMHeev1ERs31202bLQQk+31hV/R2HpYQ/1M6uT1jkg5y0UPR
wFX16yU7nM+OblGIw9fsASF0nxZw+fO7d2X2yh1PakfvF2wbrP27C/sgJmOw9upf
sBxEc40Xe0d5FMh0YKQASAhfZvKfl8nyQR2Drmm2x/33QQo15QyRvjkeWQIDQAAB
oAAwDQYJKoZIhvcNAQEBBQADgYEALM90r4d79X6vXld0qjuYXfBC0v4FrPfr
ABS/y6CmNvVysWBUchXYIGTwf4I4+s0618gYfoPUNL/L82djS18TLrUr6wpCOs
Q7APApx7WN1e4iz0fjAUU+C71NcobCAhwF1o6qzNEjpQ/2yfK9C7shb3S2JZBfe
ewItyCo=

-----END CERTIFICATE REQUEST-----

The following example shows the redundancy, show, and serial-number keywords in the crypto pki server command.

Router(config)# crypto pki server MYCA
Router(cs-server)# grant auto
Router(cs-server)# redundancy
Router(cs-server)# serial-number 0x4c
Router(cs-server)# show redundancy
serial-number 0x4C
  grant auto
end

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>crypto pki server info</td>
<td>Displays all outstanding certificate enrollment requests.</td>
</tr>
<tr>
<td></td>
<td>ip http server</td>
<td>Enables an HTTP server on your network.</td>
</tr>
</tbody>
</table>
crypto pki server grant

To grant all or certain simple certificate enrollment protocol (SCEP) requests, use the crypto pki server grant command in privileged EXEC mode.

```
crypto pki server cs-label grant { all | req-id }
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs-label</td>
<td>Name of the certificate server. The name must match the name specified via</td>
</tr>
<tr>
<td></td>
<td>the crypto pki server command.</td>
</tr>
<tr>
<td>all</td>
<td>All certificate enrollment requests are granted.</td>
</tr>
<tr>
<td>req-id</td>
<td>ID associated with a specific enrollment request in the enrollment request</td>
</tr>
<tr>
<td></td>
<td>database. Use the crypto pki server info requests command to display the ID.</td>
</tr>
</tbody>
</table>

**Defaults**

If this command is not issued, the certificate server keeps the requests in a pending state.

**Command Modes**

Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

After you enable the crypto pki server grant command, your certificate server will immediately grant all specified certificate requests. Certificate requests that are not granted will expire after the time that was specified using the lifetime enrollment-request command.

**Examples**

The following example shows to grant all manual enrollment requests for the certificate server “mycs”:

```
Router# crypto pki server mycs grant all
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server</td>
</tr>
<tr>
<td></td>
<td>configuration mode.</td>
</tr>
<tr>
<td>crypto pki server reject</td>
<td>Rejects all or certain SCEP requests.</td>
</tr>
</tbody>
</table>
crypto pki server info crl

**Note**
Effective with Cisco IOS Release 12.4(20)T, the `crypto pki server info crl` command is replaced by the `show crypto pki server crl` command. See the `show crypto pki server crl` command for more information.

To display information regarding the status of the current certificate revocation list (CRL), use the `crypto pki server info crl` command in privileged EXEC mode.

```plaintext
crypto pki server cs-label info crl
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cs-label</code></td>
<td>Name of the certificate server. The name must match the name specified via the <code>crypto pki server</code> command.</td>
</tr>
</tbody>
</table>

**Defaults**
No default behavior or values

**Command Modes**
Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(20)T</td>
<td>This command was replaced by the <code>show crypto pki server crl</code> command.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**
CRLs are issued once every specified time period via the `lifetime crl` command. It is the responsibility of the network administrator to ensure that the CRL is available from the location that is specified via the `cdp-url` command. To access information, such as the lifetime and location of the CRL, use the `crypto pki server info crl` command.

**Examples**
The following example shows how to access CRL information for the certificate server “mycs”:

```plaintext
Router# crypto pki server mycs info crl
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cdp-url</code></td>
<td>Specifies a CDP to be used in certificates that are issued by the certificate server.</td>
</tr>
<tr>
<td><code>crypto pki server</code></td>
<td>Enables a Cisco IOS certificate server and enter certificate server configuration mode.</td>
</tr>
<tr>
<td><code>lifetime crl</code></td>
<td>Defines the lifetime of the CRL that is used by the certificate server.</td>
</tr>
</tbody>
</table>
crypto pki server info requests

Note

Effective with Cisco IOS Release 12.4(20)T, the crypto pki server info requests command is replaced by the show crypto pki server requests command. See the show crypto pki server requests command for more information.

To display all outstanding certificate enrollment requests, use the crypto pki server info requests command in privileged EXEC mode.

crypto pki server cs-label info requests

Syntax Description

cs-label

Name of the certificate server. The name must match the name specified via the crypto pki server command.

Defaults

No default behavior or values

Command Modes

Privileged EXEC

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(2)T</td>
<td>The command output was modified to include shadow CA certificate information.</td>
</tr>
<tr>
<td>12.4(20)T</td>
<td>This command was replaced by the show crypto pki server requests command.</td>
</tr>
</tbody>
</table>

Usage Guidelines

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 show pki server command for a complete list of certificate enrollment request states.)
  - The certificate server refers to the command-line interface (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 Simple Certificate Enrollment Protocol (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.
  - Forwards to the request to the certification authority (CA) core, where it will generate and sign the appropriate certificate, store the certificate in the enrollment request database, and return the request to the built-in certificate server SCEP server, who will reply to the end user with the certificate on the next SCEP request.
If the connection of the client has closed, the certificate server will wait for client user to request another certificate.

All enrollment requests transitions through the certificate enrollment states that are defined in Table 27.

### Table 27 Certificate Enrollment Request State Descriptions

<table>
<thead>
<tr>
<th>Certificate Enrollment State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td>The request has been created by the SCEP server.</td>
</tr>
<tr>
<td>authorized</td>
<td>The certificate server has authorized the request.</td>
</tr>
<tr>
<td>malformed</td>
<td>The certificate server has determined that the request is invalid for cryptographic reasons.</td>
</tr>
<tr>
<td>denied</td>
<td>The certificate server has denied the request for policy reasons.</td>
</tr>
<tr>
<td>pending</td>
<td>The enrollment request must be manually accepted by the network administrator.</td>
</tr>
<tr>
<td>granted</td>
<td>The CA core has generated the appropriate certificate for the certificate request.</td>
</tr>
</tbody>
</table>

### Examples

The following example shows output for the certificate server “certsrv1,” which has a pending certificate enrollment request:

```
Router# crypto pki server certsrv1 info requests
Enrollment Request Database:
ReqID State Fingerprint SubjectName
-----------------------------------------------
1 pending 0A71820219260E526D250ECC59857C2D serialNumber=2326115A+hostname=831.
```

The following example shows the output for shadow PKI certificate info requests:

```
Router# crypto pki server mycs 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 A426AF07FE3A4BB69062E0E47198E58F hostname=client
```

Router rollover certificates requests:
<table>
<thead>
<tr>
<th>ReqID</th>
<th>State</th>
<th>Fingerprint</th>
<th>SubjectName</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>pending</td>
<td>B69062E0E47198E5BFA426AF07FE3A4B</td>
<td>hostname=client</td>
</tr>
</tbody>
</table>

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>crypto pki server</strong></td>
<td>Enables a Cisco IOS certificate server and enters PKI configuration mode.</td>
</tr>
</tbody>
</table>
crypto pki server password generate

To generate a password for simple certificate enrollment protocol (SCEP) requests that can be used only one time, use the `crypto pki server password generate` command in privileged EXEC mode.

```
crypto pki server cs-label password generate [minutes]
```

**Syntax Description**

- `cs-label`: Name of the certificate server. The name must match the name specified via the `crypto pki server` command.
- `minutes`: (Optional) Length of time, in minutes, that the password is valid. Valid times range from 1 to 1440 minutes. The default value is 60 minutes.

**Defaults**

If this command is not enabled, no password is created.

**Command Modes**

Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

SCEP, which is the only supported enrollment protocol, supports two client authentication mechanisms—manual and preshared key. Manual enrollment requires the administrator at the certification authority (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.

**Note**

Only one password is valid at a time; if a second password is generated, the previous password is no longer valid.

**Examples**

The following example shows how to generate a one-time password that is valid for 75 minutes for the certificate server “mycs”:

```
Router# crypto pki server mycs password generate 75
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
</tbody>
</table>
crypto pki server reject

To reject all or certain Simple Certificate Enrollment Protocol (SCEP) requests, use the `crypto pki server reject` command in privileged EXEC mode.

```
crypto pki server cs-label reject {all | req-id}
```

**Syntax Description**

- **cs-label**: Name of the certificate server. The name must match the name specified via the `crypto pki server` command.
- **all**: All certificate enrollment requests are rejected.
- **req-id**: ID associated with a specific enrollment request in enrollment request database. Use the `crypto pki server info requests` command to display the ID.

**Defaults**

If this command is not issued, the certificate server keeps the requests in a pending state.

**Command Modes**

Privileged EXEC

**Command History**

- **Release**: 12.3(4)T
- **Modification**: This command was introduced.

**Usage Guidelines**

After you enable the `crypto pki server reject` command, your certificate server will immediately reject all certificate requests.

SCEP, which is the only supported enrollment protocol, supports two client authentication mechanisms—manual and preshared key. Manual enrollment requires the administrator at the certification authority (CA) server to specifically authorize the enrollment requests. The administrator can become overloaded if there are numerous enrollment requests. Thus, the `crypto pki server reject` command can be reduce user interaction by automatically rejecting all or specific enrollment requests.

**Examples**

The following example shows how reject all manual enrollment requests for the certificate server “mycs”:

```
Router# crypto pki server mycs reject all
```
crypto pki server remove

To remove enrollment requests that are in the certificate server Enrollment Request Database, use the crypto pki server remove command in privileged EXEC mode. This command does not have a no form.

```
crypto pki server cs-label remove {all | req-id}
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs-label</td>
<td>Name of the certificate server.</td>
</tr>
<tr>
<td>all</td>
<td>Removes all enrollment requests.</td>
</tr>
<tr>
<td>req-id</td>
<td>Removes the specified enrollment request.</td>
</tr>
</tbody>
</table>

**Defaults**

Enrollment requests will remain in the certificate server database.

**Command Modes**

Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(11)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

After the certificate server receives an enrollment request, it can leave the request in pending, reject it, or grant it. Before this command was added, the request would be left in the Enrollment Request Database for 1 hour until the client polled the certificate server for the result of the request. This command allows you to remove individual or all requests from the database, especially useful if the client leaves and never polls the certificate server.

In addition, the use of this command also allows the server to be returned to a clean slate with respect to the keys and transaction IDs. Thus, it is a useful command to use during troubleshooting with a Simple Certificate Enrollment Protocol (SCEP) client that may be behaving badly.

**Examples**

The following example shows that all enrollment requests are to be removed from the certificate server:

```
Router# enable
Router# crypto pki server server1 remove all
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server info request</td>
<td>Displays all outstanding enrollment requests.</td>
</tr>
</tbody>
</table>
crypto pki server request pkcs10

To manually add a certificate request to the request database, use the `crypto pki server request pkcs10` command in privileged EXEC mode.

```
crypto pki server cs-label request pkcs10 {url | scep | terminal} [base64 | pem | hex [transaction-id [nonce [request-id]]]]
```

### Syntax Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cs-label</code></td>
<td>Name of the certificate server. The name must match the name specified through the <code>crypto pki server</code> command.</td>
</tr>
<tr>
<td><code>url</code></td>
<td>URL of the file systems from which the certificate server should retrieve the PKCS10 enrollment request and to which it should post the granted certificate.</td>
</tr>
<tr>
<td><code>scep</code></td>
<td>Specifies the certificate is returned using Secure Certificate Enrollment Protocol (SCEP) request.</td>
</tr>
<tr>
<td><code>terminal</code></td>
<td>Certificate requests is manually pasted from the console terminal, and the granted certificate is displayed on the console.</td>
</tr>
<tr>
<td><code>base64</code></td>
<td>(Optional) Specifies the certificate is returned without privacy-enhanced mail (PEM) headers, regardless of whether PEM headers were used in the request.</td>
</tr>
<tr>
<td><code>pem</code></td>
<td>(Optional) Specifies the certificate 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.</td>
</tr>
<tr>
<td><code>hex</code></td>
<td>(Optional) Specifies the certificate is returned in hexadecimal. Pending requests will also be synchronized with the standby certificate server in hexadecimal.</td>
</tr>
<tr>
<td><code>transaction-id</code></td>
<td>(Optional) Transaction ID in hexadecimal format.</td>
</tr>
<tr>
<td><code>nonce</code></td>
<td>(Optional) Nonce word in hexadecimal format.</td>
</tr>
<tr>
<td><code>request-id</code></td>
<td>(Optional) Request ID. Valid values are from 1 to 999.</td>
</tr>
</tbody>
</table>

### Command Default

None

### Command Modes

Privileged EXEC (#)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.3(11)T</td>
<td>This command was modified. The <code>pem</code> keyword was added.</td>
</tr>
</tbody>
</table>
Use the **crypto pki server request pkcs10** command to manually add a base64-encoded, PEM-formatted, or hexadecimal-encoded PKCS10 certificate enrollment request. This command is especially useful when the client does not have a network connection with the certificate server so that it can do Simple Certificate Enrollment Protocol (SCEP) enrollment. After the certificate is granted, the certificate will be displayed on the console terminal using base64 encoding if the **terminal** keyword is specified, or it will be sent to the file system that is specified using the **url** argument.

The **url** argument allows you to specify or change the location in which the certificate server retrieves the new certificate request and posts the granted certificate.

### Examples

The following example shows how to manually add a base64-encoded certificate request with PEM boundaries to the request database:

```
Router# crypto pki server mycs request pkcs10 terminal pem
% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIBdTCB3wIBADA2MQswCQYDVQQGEwJVUzEHAMQGA1UEChMNQ2lzY28gU3lzdGVt
czEPMA0GA1UEAxMDGdvzdBAAABAAAAMIGfMA0GCSqGSIb3DQEBAQUA4GNADCBiQKBgQDF
EFukc21CFSH0jD6hFR2n8pdpnLYwc0m68N3RY9084730/96utTHv2qEE0
rW97bMQZkEBLhVLC09T5KvHUbilHqM/w/BqgV8IWHZzskH/UM8kd67Vkjjb
fe+F7W7rF104vR4zFtP2P3FZ+AZ29bucz9s8q9nQIDAAbObAwDQYJKoZIhvcN
AQEEBQAdqYEAQUGCMzAJNWB0CwEmG8E0FSZWDmFictm8NVaZYMPlc+v6lwpK
RmtD1k91Vw/qT5FJm8L/m0w+TqgTrm2dgsd5t54dx0dg5G5E0a4
T302paxER1/gqJ6p80D7jF9gvdskRziCpyD4/JfblWhSVQzmvYXaxVQ=
-----END CERTIFICATE REQUEST-----
% Enrollment request pending, reqId=2
```

```
Router# crypto pki server mycs request pkcs10 terminal pem
% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIBdTCB3wIBADA2MQswCQYDVQQGEwJVUzEHAMQGA1UEChMNQ2lzY28gU3lzdGVt
czEPMA0GA1UEAxMDGdvzdBAAABAAAAMIGfMA0GCSqGSIb3DQEBAQUA4GNADCBiQKBgQDF
EFukc21CFSH0jD6hFR2n8pdpnLYwc0m68N3RY9084730/96utTHv2qEE0
rW97bMQZkEBLhVLC09T5KvHUbilHqM/w/BqgV8IWHZzskH/UM8kd67Vkjjb
fe+F7W7rF104vR4zFtP2P3FZ+AZ29bucz9s8q9nQIDAAbObAwDQYJKoZIhvcN
AQEEBQAdqYEAQUGCMzAJNWB0CwEmG8E0FSZWDmFictm8NVaZYMPlc+v6lwpK
RmtD1k91Vw/qT5FJm8L/m0w+TqgTrm2dgsd5t54dx0dg5G5E0a4
T302paxER1/gqJ6p80D7jF9gvdskRziCpyD4/JfblWhSVQzmvYXaxVQ=
-----END CERTIFICATE REQUEST-----

```

**Usage Guidelines**

Use the **crypto pki server request pkcs10** command to manually add a base64-encoded, PEM-formatted, or hexadecimal-encoded PKCS10 certificate enrollment request. This command is especially useful when the client does not have a network connection with the certificate server so that it can do Simple Certificate Enrollment Protocol (SCEP) enrollment. After the certificate is granted, the certificate will be displayed on the console terminal using base64 encoding if the **terminal** keyword is specified, or it will be sent to the file system that is specified using the **url** argument.

The **url** argument allows you to specify or change the location in which the certificate server retrieves the new certificate request and posts the granted certificate.

**Examples**

The following example shows how to manually add a base64-encoded certificate request with PEM boundaries to the request database:

```
Router# crypto pki server mycs request pkcs10 terminal pem
% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIBdTCB3wIBADA2MQswCQYDVQQGEwJVUzEHAMQGA1UEChMNQ2lzY28gU3lzdGVt
czEPMA0GA1UEAxMDGdvzdBAAABAAAAMIGfMA0GCSqGSIb3DQEBAQUA4GNADCBiQKBgQDF
EFukc21CFSH0jD6hFR2n8pdpnLYwc0m68N3RY9084730/96utTHv2qEE0
rW97bMQZkEBLhVLC09T5KvHUbilHqM/w/BqgV8IWHZzskH/UM8kd67Vkjjb
fe+F7W7rF104vR4zFtP2P3FZ+AZ29bucz9s8q9nQIDAAbObAwDQYJKoZIhvcN
AQEEBQAdqYEAQUGCMzAJNWB0CwEmG8E0FSZWDmFictm8NVaZYMPlc+v6lwpK
RmtD1k91Vw/qT5FJm8L/m0w+TqgTrm2dgsd5t54dx0dg5G5E0a4
T302paxER1/gqJ6p80D7jF9gvdskRziCpyD4/JfblWhSVQzmvYXaxVQ=
-----END CERTIFICATE REQUEST-----

```

**Release**

**Modification**

<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(6)T</td>
<td>This command was modified. The <strong>base64</strong> keyword was added.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified. The <strong>scep</strong> and <strong>hex</strong> keywords and <strong>transaction-id</strong>, <strong>nonce</strong>, and <strong>request-id</strong> arguments were added. Command accepts the PKCS10 certificate and the signing certificate in hexadecimal as well as in base64 encoding.</td>
</tr>
<tr>
<td>15.1(1)T</td>
<td>This command was modified. The prompt for entering a certificate in hex mode has changed from config-pubkey to config-pki-hexmode.</td>
</tr>
</tbody>
</table>
The following example shows how to retrieve a certificate request and add it to the request database (using the url argument):

Note

The request file name should have a “.req” extension and the certificate file name a “.crt” extension.

Router# crypto pki server mycs request pkcs10 tftp://192.0.2.129/router5
% Retrieving Base64 encoded or PEM formatted PKCS10 enrollment request...
Reading file from tftp://192.0.2.129/router5.req
Loading router5.req from 192.0.2.129 (via Ethernet0): ![OK - 582 bytes]
% Enrollment request pending, reqId=1

Router# crypto pki server mycs grant 1
% Writing out the granted certificate...
!Writing file to tftp://192.0.2.129/router5.crt!

The following example shows how to manually add a hexadecimal-encoded certificate request with PEM boundaries to the request database in Cisco IOS Release 15.0(1)M and earlier:

Router# crypto pki server mycs request pkcs10 scep hex 0C4A3A2CA5C2E6DDCD740A4259759E2
5811E7CB133BAC936EF48C6187F4AD22 3
PKCS10 request in hex
Enter the PKCS10 in hexadecimal representation....

Router(config-pubkey)# 3082010E 3081B902 0100301D 311B3019 06092A86 4886F70D 01000216
0C697073
Router(config-pubkey)# 6563662D 33383435 61305C30 0D06092A 864886F7 0D010101 0500034B
00304802
Router(config-pubkey)# 4100B660 EF764AD6 A896E03E 0D1A1A16 5450857C 9B2CC04E B61719E5
2216CBF2
Router(config-pubkey)# 1973B464 17E78829 22CDBB07 FBD015F1 2A0A8DD7 5396EAA1 A2A65132
91246662
Router(config-pubkey)# 62C90203 010001A0 37301406 092A8648 86F70D01 09073107 13056369
73636F30
Router(config-pubkey)# 1F060A60 86480186 F8450109 08311104 0F300D30 0B060355 10DF0404
030205A0
Router(config-pubkey)# 300D0609 2A864886 F70D0101 04050003 41062A5 81B4C7F2 BDCEE03D
998BAD2B
Router(config-pubkey)# 1E763461 EBB812EB 4082E2BB 273AA5DD 74FF7E12 E16035E9 4525A041
AF65E48F
Router(config-pubkey)# F0B6E13C 2646F943 5C23A634 BC508C1F 343A
Router(config-pubkey)# 30820133 3081CE02 0101300D 06092A86 4886F70D 01010405 00301D31
1B301906
Router(config-pubkey)# 092A8648 86F70D01 0902160C 69707365 63662D33 3B343561 301E170D
30393031
Router(config-pubkey)# 31323032 33323039 5A170D31 39303131 30303233 3230395A 301D311B
30190609
Router(config-pubkey)# 97F8333 5DDA951
Router(config-pubkey)# quit
Enter the certificate in hexadecimal representation....

Router(config-pubkey)# quit

The following example shows how to manually add a hexadecimal-encoded certificate request with PEM boundaries to the request database in Cisco IOS Release 15.1(1)T and later:

Router# crypto pki server mycs request pkcs10 scep hex 0C4A3A2CA5C2E6DDCD740A4259759E2
5811E7CB133BAC936EF48C6187F4AD22 3
PKCS10 request in hex
Enter the PKCS10 in hexadecimal representation....
Router (config-pki-hexmode) \# 3082010E 3081B902 0100301D 311B3019 06092A86 4886F70D 01090216 0C697073
Router (config-pki-hexmode) \# 6563662D 33383435 61305C30 0D06092A 864886F7 0D010101 0500034B 00304802
Router (config-pki-hexmode) \# 41008660 EF764A66 A896E03E 0D1A1A16 5450857C 9B2CC04E B61719E5 2216CBF2
Router (config-pki-hexmode) \# 62C90203 010001A0 37301406 092A8648 86F70D01 09073107 13056369 73636F30
Router (config-pki-hexmode) \# 1F060A60 86480186 F70D0101 04050003 410062A5 81B4C7F2 BDCEE03D 998BAD2B
Router (config-pki-hexmode) \# 1E763461 EBB812EB 4082E2BB 273AA5DD 74FF7E12 E16035E9 4525A041 AP65E48F
Router (config-pki-hexmode) \# FOE613C 2646F943 5C23A634 BC50BC1F 343A
Router (config-pki-hexmode) \# 30820123 3081CE02 0101300D 06092A86 4886F70D 01010405 00301D31 1B301906
Router (config-pki-hexmode) \# 092A8648 86F70D01 0902160C 69707365 63662D33 38343561 301E170D 30393031
Router (config-pki-hexmode) \# 313323032 33323039 5A170D31 39303131 30303323 3230395A 301D311B 30190609
Router (config-pki-hexmode) \# 2A864886 F70D0109 02160C69 70736563 662D3338 34356130 5C300D06 092A8648
Router (config-pki-hexmode) \# 6F70D01 01010500 03480030 48024100 B660EF76 4AD6A896 E03E0D1A 1A165450
Router (config-pki-hexmode) \# 857C9B2C C04EB617 19E52216 CBF21973 B46417E7 882922CD BD87FBD0 15F12A0A
Router (config-pki-hexmode) \# 8DD75336 EAA1A2A6 51329124 66D262C9 02030100 01300D06 092A8648 86F70D01
Router (config-pki-hexmode) \# 01040500 03410041 B2EBC43A 7F5FD62A DBA8574 655D0C5D 84CC7B5 48643525
Router (config-pki-hexmode) \# E85E4B06 5A65A27F 6066BC8C 552AF9FF4 CE643C66 44441BF0 053325DC 736FD696
Router (config-pki-hexmode) \# 97F8335 DDA951
Router (config-pki-hexmode) \# quit
Enter the certificate in hexadecimal representation....

Router (config-pki-hexmode) \# quit

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
<tr>
<td>crypto pki server grant</td>
<td>Grants all or certain SCEP requests.</td>
</tr>
<tr>
<td>show crypto pki server</td>
<td>Displays the current state and configuration of a certificate server.</td>
</tr>
</tbody>
</table>
crypto pki server revoke

To revoke a certificate on the basis of its serial number, use the crypto pki server revoke command in privileged EXEC mode.

    crypto pki server cs-label revoke certificate-serial-number

Syntax Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs-label</td>
<td>Name of the certificate server. The name must match the name specified via</td>
</tr>
<tr>
<td></td>
<td>the crypto pki server command.</td>
</tr>
<tr>
<td>certificate-serial-number</td>
<td>Serial number of the certificate that is to be revoked. The serial number</td>
</tr>
<tr>
<td></td>
<td>can be a hexadecimal number with the prefix “0x” (for example, 0x4c) or a</td>
</tr>
<tr>
<td></td>
<td>decimal number (for example, 76).</td>
</tr>
</tbody>
</table>

Defaults

Certificates are revoked on the basis of their name.

Command Modes

Privileged EXEC

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>The command was modified to remove the serial-number check against the last-issued serial number.</td>
</tr>
</tbody>
</table>

Usage Guidelines

When a new certificate revocation list (CRL) is issued, the certificate server obtains the previous CRL, makes the appropriate changes, and resigns the new CRL. A new CRL is issued after a certificate is revoked from the CLI. If this process negatively affects router performance, the crypto pki server revoke command can be used to revoke a list or range of certificates.

Note

In Cisco IOS Release 15.0(1)M, the serial number to be revoked is not compared with the last-issued serial number.

Note

A new CRL cannot be issued unless the current CRL is revoked or changed.

Examples

The following examples show how to revoke a certificate with the serial number 76 (for example, 0x4c in hexadecimal) from the certificate server “mycs”:

    Router# crypto pki server mycs revoke 76
    Router# crypto pki server mycs revoke 0x4c

Related Commands
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdp-url</td>
<td>Specifies that CDP should be used in the certificates that are issued by the certificate server.</td>
</tr>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
</tbody>
</table>
crypto pki server start

To enable a Cisco IOS certificate server, use the `crypto pki server start` command in privileged EXEC mode. To disable a certificate server, use the `crypto pki server stop` command.

```
crypto pki server servername start
```

**Syntax Description**

<table>
<thead>
<tr>
<th><code>servername</code></th>
<th>Name of the certificate server.</th>
</tr>
</thead>
</table>

**Note**

The certificate server name must not exceed 13 characters.

**Command Default**

The certificate server is disabled.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.0(1)M</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Using the `crypto pki server start` command is the same as using the `no shut` command in DSP configuration mode.

**Examples**

The following example shows how to enable a certificate server on a router:

```
Router# crypto pki server MYCA start
% 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:
Re-enter password:
% Certificate Server enabled.
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>crypto pki server stop</code></td>
<td>Disables a Cisco IOS certificate server.</td>
</tr>
<tr>
<td><code>show crypto pki server</code></td>
<td>Displays the current state and configuration of a certificate server.</td>
</tr>
</tbody>
</table>
crypto pki server stop

To disable a Cisco IOS certificate server, use the **crypto pki server stop** command in privileged EXEC mode.

```
crypto pki server servername stop
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>servername</code></td>
<td>Name of the certificate server.</td>
</tr>
</tbody>
</table>

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.0(1)M</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Using the **crypto pki server stop** command is the same as using the **shutdown** command in DSP configuration mode.

**Examples**

The following example shows how to disable a certificate server:

```
Router# crypto pki server MYCA stop
Certificate server 'shut' event has been queued for processing.
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>crypto pki server start</strong></td>
<td>Enables a Cisco IOS certificate server.</td>
</tr>
<tr>
<td><strong>show crypto pki server</strong></td>
<td>Displays the current state and configuration of a certificate server.</td>
</tr>
</tbody>
</table>
crypto pki server trim

To trim certificates from the certificate revocation list (CRL), use the crypto pki server trim command in privileged EXEC mode.

```
crypto pki server [cs-label] trim {expired [start-number [end-number]] [verbose]} | generate expired-list [start-number end-number] [url url] | url url [verbose]
```

Syntax Description

- **cs-label**: Name of the certificate server. The name must match the name specified using the **crypto pki server** command.
- **expired**: Specifies that the expired certificates are to be trimmed from the CRL.
- **start-number**: The beginning of the certificate serial number range to check and trim from the CRL if the certificate has expired.
- **end-number**: (Optional) The ending number of the certificate serial number range to check and trim from the CRL if the certificate has expired.
- **verbose**: Displays information about the action taken on the certificates checked in the CRL.
- **generate**: Generates information about CRL trimming.
- **expired-list**: Generates information about trimmed expired certificates.
- **url url**: Specifies the location of the expired certificate list, which contains a list of certificate serial numbers to be trimmed from the CRL.

Command Default

All certificates in the specified certificate server database will be searched to locate and to trim expired certificates.

Command Modes

Privileged EXEC (#)

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(20)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified in a release earlier than Cisco IOS Release 15.0(1)M. The <strong>generate</strong> keyword was added.</td>
</tr>
</tbody>
</table>

Usage Guidelines

This command trims expired certificates from the CRL. Only certificates that are expired and have accurate and complete information in the certificate database can be trimmed from the database. Depending on the size and location of the certificate database, searching the database for expired certificates may be a time-consuming process. Depending on your environment, you may choose one of three methods to search and to trim your CRL:

- Search the entire certificate database.
  
  This is usually the most time-consuming and resource-consuming method.

- Specify a range of certificate serial numbers to search.
If a large number of certificates are in your certificate database or if your certificate database is stored at a remote location (for example, TFTP or Secure Copy [SCP]) you may limit the range of certificates to search by specifying both the starting and ending certificate serial numbers. If no starting and ending certificate serial numbers are specified, the entire certificate database will be searched and all expired certificates will be trimmed.

- Use an input list to specify the expired certificates to be trimmed from the CRL.

This is the most scalable method because it divides the process into two steps: searching the certificate database for expired certificates and trimming the CRL. An input file listing expired certificate serial numbers may be generated using a Perl script or similar program, manually, or by issuing the `crypto pki server trim generate expired-list` command. The input list must follow the format as shown:

```
# CRL Trimming file generated on 01/31/2008
version=1
35
37
```

Lines that begin with a pound sign (#) are inserted comments. The second line contains a version string indicating the file type. Each remaining line (in this example lines 35 and 37) contains a certificate serial number indicating one certificate to be removed from the CRL.

**Examples**

The following example shows how to check and trim the CRL of all expired certificates in the certificate database for the certificate server “mycs”:

Router# crypto pki server mycs trim expired

The following example shows how to check and trim the CRL of expired certificates within the certificate serial number range 0x1–0x3 in the certificate database for the certificate server “mycs”. The result is the same as generating and using an input file of expired certificate serial numbers, as shown in the next example.

Router# crypto pki server mycs trim expired 0x1 end 0x3

The following example shows how to generate a list of expired certificate serial numbers, store the list on an HTTP server, then use the resulting list to trim the CRL of all expired certificates for the certificate server “mycs”:

Router# crypto pki server mycs trim generate expired-list 0x1 0x3 url http://databaselocation/expired-certs.1st

Router# crypto pki server mycs trim url http://databaselocation/expired-certs.1st

The following example shows how to check and trim the CRL for only one certificate serial number in the certificate database for the certificate server “mycs.” If the certificate with the serial number 45 has expired, it will be trimmed from the CRL.

Router# crypto pki server mycs trim expired 0x2

The following example shows how to trim the CRL of all expired certificates for the certificate server “mycs” and display the resulting action taken for each certificate serial number:

Router# crypto pki server mycs trim expired verbose

Certificate 2: Expired. Removed from CRL.
Certificate F4240: Expired. Removed from CRL.
Certificate 4593: Not Removed.
Certificate 1234: Not Removed.

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
<tr>
<td>crypto pki server trim generate expired-list</td>
<td>crypto pki server trim generate expired-list</td>
<td>Generates a list of expired certificates in the CRL.</td>
</tr>
</tbody>
</table>
**crypto pki server trim generate expired-list**

To generate a list of expired certificates in the current certificate revocation list (CRL), use the `crypto pki server trim generate expired-list` command in privileged EXEC mode.

```
crypto pki server cs-label trim generate expired-list [start number end number] [url url]
```

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cs-label</code></td>
<td>Name of the certificate server. The name must match the name specified via the <code>crypto pki server</code> command.</td>
</tr>
<tr>
<td><code>start number</code></td>
<td>(Optional) The first certificate serial number from which to begin searching the CRL for expired certificates. To locate expired certificates within a range both the starting certificate serial number and the ending certificate serial number must be specified.</td>
</tr>
<tr>
<td><code>end number</code></td>
<td>(Optional) The last certificate serial number that will be checked when searching the CRL for a range of expired certificates.</td>
</tr>
<tr>
<td><code>url url</code></td>
<td>(Optional) Specifies the location where the resulting list of expired certificates will be stored.</td>
</tr>
</tbody>
</table>

**Command Default**

All certificates in the specified certificate server database will be searched to locate expired certificates.

**Command Modes**

Privileged EXEC (#)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(20)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command generates a list of expired certificates that are in the CRL for the specified certificate server. The resulting list of expired certificates may be used as input to the `crypto pki server trim` command to remove the listed certificates from the CRL resulting in trimming, or revoking, the expired certificates.

Only certificates that have accurate and complete information in the certificate database can be automatically added to the list of expired certificates and later trimmed from the database. Only CRL entries for expired certificates can be trimmed.

If there are a large number of certificates in your certificate database or if your certificate database is stored at a remote location, for example TFTP or SCP, you may limit the range of certificates to search by specifying both the starting and ending certificate serial numbers. If no starting and ending certificate serial numbers are specified, the entire certificate database will be searched and all expired certificates will be added to the expired certificates list.

A URL may be specified to save the list of expired certificates to a specified location. If no URL is specified, the list of expired certificates will be printed on your terminal. The list may then be cut and pasted to a file.
Examples

The following example shows both how to generate a list of expired certificates within the certificate serial number range 34–38 in the certificate database for the certificate server “mycs” and how to save the resulting list to an HTTP location:

Router# crypto pki server mycs trim generate expired-list start 34 end 38 url http://databaselocation/expired-certs.1st

The following example shows the resulting list of expired certificates in the file expired-certs.1st:

# CRL Trimming file generated on 01/31/2008
version=1
35
37

Lines that begin with a pound sign (#) are inserted comments. The second line contains a version string indicating the file type. Each remaining line, in this example lines 35 and 37, contains a certificate serial number indicating one certificate to be removed from the CRL.

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
<tr>
<td>crypto pki server trim</td>
<td>Trims certificates from the certificate revocation list.</td>
</tr>
</tbody>
</table>
crypto pki server unrevoke

To recover a revoked certificate, that is to remove a certificate from the certificate revocation list (CRL), use the `crypto pki server unrevoke` command in privileged EXEC mode.

```
crypto pki server cs-label unrevoke certificate-serial-number
```

### Syntax Descriptions

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cs-label</code></td>
<td>Name of the certificate server. The name must match the name specified via the <code>crypto pki server</code> command.</td>
</tr>
<tr>
<td><code>certificate-serial-number</code></td>
<td>Serial number of the certificate that is to be recovered. The serial number can be a hexadecimal number with the prefix “0x” (for example, 0x4c) or a decimal number (for example, 76).</td>
</tr>
</tbody>
</table>

### Command Default

None.

### Command Modes

Privileged EXEC (#)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(20)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

If a certificate is erroneously revoked, either the client has to reenroll in the PKI or the administrator may recover the revoked certificate by issuing the `crypto pki server unrevoke` command. This command removes a certificate, specified by its serial number, from the CRL. The CRL is then resigned and can be republished.

### Examples

The following examples show how to unrevoke a certificate with the serial number 76, or 0x4c in hexadecimal, from the certificate server “mycs”:

```
Router# crypto pki server mycs unrevoke 76
Router# crypto pki server mycs unrevoke 0x4c
```

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>crypto pki server</code></td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
<tr>
<td><code>crypto pki server revoke</code></td>
<td>Revokes a certificate based on its serial number.</td>
</tr>
</tbody>
</table>
crypto pki token change-pin

To change the user PIN on the USB eToken, use the `crypto pki token change-pin` command in privileged EXEC mode.

```
crypto pki token token-name [admin] change-pin [pin]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token-name</td>
<td>Name of USB token specified via the <code>crypto pki token login</code> command.</td>
</tr>
<tr>
<td>admin</td>
<td>(Optional) The router will change the administrative PIN on the USB token.</td>
</tr>
<tr>
<td>pin</td>
<td>(Optional) User PIN required to access the etoken.</td>
</tr>
</tbody>
</table>

**Command Default**

None

**Command Modes**

Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

If you want to change the administrative PIN on the token, you must be logged into the eToken as an `admin` via the `crypto pki token admin login` command.

After the user PIN has been changed, you must reset the login failure count to zero (via the `crypto pki token max-retries` command). The maximum number of allowable login failures is set (by default) to 15.

**Examples**

The following example shows that the user PIN was changed to 1234:

```
crypto pki token usbtoken0 admin login 5678
crypto pki token usbtoken0 change-pin 1234
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token login</td>
<td>Logs into the USB eToken.</td>
</tr>
<tr>
<td>crypto pki token max-retries</td>
<td>Sets the maximum number of allowed failed login attempts.</td>
</tr>
</tbody>
</table>
crypto pki token encrypted-user-pin

To encrypt a USB token PIN that is stored in private NVRAM, use the **crypto pki token encrypted-user-pin** command in global configuration mode. To decrypt the token’s PIN, use the **no** form of this command.

```plaintext
crypto pki token {token-name | default} encrypted-user-pin [write] [passphrase passphrase]

no crypto pki token {token-name | default} encrypted-user-pin [write] [passphrase passphrase]
```

### Syntax Description

<table>
<thead>
<tr>
<th>token-name</th>
<th>Name of the token that will have its PIN encrypted.</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>Configures default values for tokens.</td>
</tr>
<tr>
<td>write</td>
<td>(Optional) Writes to memory immediately after the passphrase is entered. This keyword saves the running configuration to NVRAM.</td>
</tr>
<tr>
<td>passphrase passphrase</td>
<td>(Optional) Enables noninteractive command-line interface (CLI). If you do not issue this keyword, you will automatically be prompted for the passphrase.</td>
</tr>
</tbody>
</table>

**Tip**: Noninteractive CLI is provided for instances where users will not be responding to prompts, for example in scripts, configuration tools, or other automated processes.

If you are issuing this command from the console, it is recommended that you use the interactive CLI to help protect against observation from unauthorized persons.

### Command Default

The PIN stored in private NVRAM is not encrypted.

### Command Modes

Global configuration (config)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco IOS Release 12.4(11)T and implemented on 7200VXR NPE-G2 platform.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified earlier than Cisco IOS Release 15.0(1)M. The <strong>default</strong> keyword was added.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

After the token’s PIN is encrypted with the **crypto pki token encrypted-user-pin** command, no action is taken when you insert the token into the router. The user must log in to the router and enter the passphrase to decrypt the PIN before the router can use the PIN to log in to the token.

After the PIN has been successfully decrypted, the router will execute the configuration commands from the token at privilege level 15.
Tip

It is recommended that you create a passphrase different from the token’s PIN.

Also, the user should log in to the token as a “normal user” (a privilege level 1 user), so the user cannot access commands that can alter the configuration of the router.

Examples

The following example shows the configuration of a user PIN and the encryption of that user PIN:

```
! Configure the user PIN.
Router(config)# crypto pki token usbtoken0: user-pin
Enter password:
!
! Now, the user PIN can be encrypted.
!
Router(config)# crypto pki token usbtoken0: encrypted-user-pin
Enter passphrase:
Router(config)# exit
Router# show running config
.
.
  crypto pki token usbtoken0 user-pin *encrypted*
.
.
```

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token</td>
<td>Creates a PIN that automatically allows the router to log in to the USB</td>
</tr>
<tr>
<td>user-pin</td>
<td>token at router startup.</td>
</tr>
<tr>
<td>privilege</td>
<td>Configures a new privilege level for users and associates commands with that</td>
</tr>
<tr>
<td></td>
<td>privilege level.</td>
</tr>
</tbody>
</table>
crypto pki token label

To set or change the name of a USB token label, use the `crypto pki token label` command in global configuration mode.

```
crypto pki token device: label token-label
```

**Syntax Description**
- `device:` Location or name of the USB device.
- `token-label`: Specifies the label, or name, of the USB token.
  - `token-label` may be up to 31 alphanumeric characters in length, including dashes and underscores.

**Command Default**
No label is set. The USB token is known by its factory name.

**Command Modes**
Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**
After you have logged in your USB token to the router, you may want to change the factory default label. Changing the default factory name to a unique name is useful when configuring multiple USB tokens for automatic login, secondary configuration files, or other token specific settings.

**Note**
Either the device name or label may be used to specify the USB token.

If using the device name, it is followed by a colon, “:".

**Examples**
The following example shows how to change the USB token label from the “oldlabel” to “newlabel” after the token has been logged in. The router will not use the “newlabel” until the next time the token is inserted or the router is reloaded:

```
Router# configure terminal
Router(config)# crypto pki token oldlabel label newlabel
Token label changed.
```

**Related Commands**
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token user-pin</td>
<td>Creates a PIN that automatically allows the router to log into the USB token at router startup.</td>
</tr>
</tbody>
</table>
crypto pki token lock

To lock the token, use the crypto pki token lock command in privileged EXEC mode.

crypto pki token token-name lock [user-pin] [passphrase passphrase]

Syntax Description

<table>
<thead>
<tr>
<th>token-name</th>
<th>Name of the token that is to be locked.</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-pin</td>
<td>(Optional) Specifies the USB token PIN if set.</td>
</tr>
<tr>
<td>passphrase</td>
<td>(Optional) Enables the noninteractive command-line interface (CLI). If you do not issue this keyword, you will automatically be prompted for the passphrase.</td>
</tr>
</tbody>
</table>

Tip: The noninteractive CLI is provided for instances where users will not be responding to prompts, for example in scripts, configuration tools, or other automated processes.

If you are issuing this command from the console, it is recommended that you use the interactive CLI to help protect against observation from unauthorized persons.

Command Default

The token is not locked.

Command Modes

Privileged EXEC

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

Usage Guidelines

After you have locked a token with the crypto pki token lock command, all Rivest, Shamir, and Adelman (RSA) keys that have been loaded from the token will be deleted and, if configured, the secondary “unconfig” file will run with full privileges.

Examples

The following example shows how to reload a router, unlock the PIN, and then lock the PIN again:

```
Router> enable
Password:
Router# crypto pki token usbtoken0: unlock
  Token eToken is usbtoken0
!
Enter passphrase:
Token login to usbtoken0(eToken) successful
Router#
Sep 20 22:31:13.128: %CRYPTO-6-TOKENLOGIN: Cryptographic Token eToken Login Successful
```
Router# `crypto pki token usbtoken0: lock`

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>crypto pki token name</code></td>
<td>Specifies a secondary “unconfig” file.</td>
</tr>
<tr>
<td></td>
<td><code>secondary unconfig</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>file</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>crypto pki token unlock</code></td>
<td>Unlocks the token and decrypts the PIN that is stored in private NVRAM.</td>
</tr>
</tbody>
</table>
crypto pki token login

To log into the USB eToken, use the **crypto pki token login** command in privileged EXEC mode.

```plaintext
crypto pki token token-name [admin] login [pin]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token-name</td>
<td>Name of USB eToken.</td>
</tr>
<tr>
<td>admin</td>
<td>(Optional) The router will attempt to log into the token as an administrator. If this keyword is not issued, the router will attempt to log into the token as a user. <strong>Note</strong> If you want to change the PIN via the <strong>crypto pki token change-pin</strong> command, you must issue this keyword.</td>
</tr>
<tr>
<td>pin</td>
<td>(Optional) User PIN required to access the token. If a user PIN is not specified, the default PIN, 1234567890, is used.</td>
</tr>
</tbody>
</table>

**Command Default**

None

**Command Modes**

Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

This command allows you to manually log into a USB eToken. To automatically log into an eToken, issue the **crypto pki token user-pin** command, which allows you to create a PIN for automatic login.

**Examples**

The following example shows how to log into the USB eToken manually:

```plaintext
crypto pki token usbtoken0:login 1234567890
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token logout</td>
<td>Logs the router out of the USB eToken.</td>
</tr>
</tbody>
</table>
crypto pki token logout

To log the router out of the USB eToken, use the **crypto pki token logout** command in privileged EXEC mode.

**crypto pki token** token-name **logout**

**Syntax Description**

| token-name | Name of USB eToken specified via the **crypto pki token login** command. |

**Command Default**

None

**Command Modes**

Privileged EXEC

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

If you want to save any data to the USB eToken, you must log back into the eToken.

**Examples**

The following example shows how to successfully log out of a USB eToken:

```
crypto pki token usbtoken0:logout
Token eToken is usbtoken0
Token logout from usbtoken0(eToken) successful
*Jan 28 05:46:59.544:%CRYPTO-6-TOKENLOGOUT:Cryptographic Token eToken Logout Successful
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token login</td>
<td>Logs into the USB eToken.</td>
</tr>
</tbody>
</table>
crypto pki token max-retries

To set the maximum number of allowed failed login attempts, use the **crypto pki token max-retries** command in global configuration mode. To return to the default functionality (which is 15 failed login attempts), use the **no** form of this command.

```
crypto pki token {token-name | default} max-retries [number]

no crypto pki token {token-name | default} max-retries [number]
```

**Syntax Description**

- **token-name**
  - Name of USB token that the router will log into.
- **default**
  - Default value is to be used.
- **number**
  - (Optional) Number of consecutive failed login attempts the router will allow before locking out the user. Available range: 0 to 15. Default value is 15.

**Defaults**

15 failed login attempts are allowed

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

After the user PIN is changed via the **crypto pki token change-pin** command, the login failure count is automatically reset to 15; however, it is recommended that the login failure count be set to zero.

**Examples**

The following example shows how to change the allowed maximum number of failed login attempts to 20:

```
crypto pki token usbtoken0 max-retries 20
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token</td>
<td>Changes the user PIN number on the USB eToken.</td>
</tr>
<tr>
<td>change-pin</td>
<td></td>
</tr>
<tr>
<td>crypto pki token login</td>
<td>Logs into the USB eToken.</td>
</tr>
</tbody>
</table>
crypto pki token removal timeout

To set the time interval that the router waits before removing the Rivest, Shamir, and Adelman (RSA) keys that are stored in the eToken, use the `crypto pki token removal timeout` command in global configuration mode. To return to the default functionality (which is no timeout), use the `no` form of this command.

```
crypto pki token {token-name | default} removal timeout [seconds]
no crypto pki token {token-name | default} removal timeout [seconds]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>token-name</code></td>
<td>Name of USB eToken that is being removed from the router.</td>
</tr>
<tr>
<td><code>default</code></td>
<td>Default value, which is automatic RSA key removal, is to be used.</td>
</tr>
<tr>
<td><code>seconds</code></td>
<td>(Optional) Time interval, in seconds, that the router waits before removing the RSA keys and tearing down IP Security (IPSec) tunnels associated with the specified eToken. Available range: 0 to 480.</td>
</tr>
</tbody>
</table>

**Note**

If a time interval is not specified, all RSA keys and associated tunnels are immediately torn down after the eToken is removed from the router.

**Defaults**

The default timeout is zero, which causes the RSA keys to be removed automatically after the eToken is removed from the router. The default appears in the running configuration as:

```
crypto pki token default removal timeout 0
```

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

After the eToken is removed from the router, you can clear from your router any RSA keys that were obtained from the eToken; all IPSec tunnels that used those RSA keys for authentication are also torn down. Both the keys and tunnels are immediately cleared unless otherwise specified via the `crypto pki token removal timeout` command.

Although the RSA keys remain on the eToken, they can only be accessed with the correct PIN. Too many unsuccessful attempts to log into the eToken will disable the PIN and any further login attempts will be refused.

**Note**

The `no` version of this command does not remove RSA keys from the router. To immediately remove RSA keys from the router, set the timeout value to zero.
Examples

The following example shows how to set the time that the router will wait before removing the RSA keys that are stored in the eToken after the eToken has been removed from the router:

```
crypto pki token usbtoken0 removal timeout 60
```
crypto pki token secondary config

To merge a specified file with the running configuration after the eToken is logged in to the router, use the `crypto pki token secondary config` command in global configuration mode. To remove the specified file, use `no` form of the command.

```
crypto pki token {token-name | default} secondary config [file]
no crypto pki token {token-name | default} secondary config [file]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>token-name</code></td>
<td>Name of USB eToken that will have its running configuration merged with the secondary configuration file.</td>
</tr>
<tr>
<td><code>default</code></td>
<td>Sets the default values for tokens.</td>
</tr>
<tr>
<td><code>file</code></td>
<td>(Optional) Name of the file that will be merged with the running configuration.</td>
</tr>
</tbody>
</table>

**Note** The filename is relative to the eToken, so the name should not include a device name such as “usbtoken0:.”

**Command Default**

A secondary configuration file does not exist.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified earlier than Cisco IOS Release 15.0(1)M. The <code>default</code> keyword was added.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Use the `crypto pki token secondary config` command if you want to merge, not overwrite, a file with the running configuration on the router. The secondary configuration is processed after the eToken is logged in to the router.

**Examples**

The following example shows how to merge the secondary configuration file “CONFIG1.CFG” with the current running configuration:

```
Router# configure terminal
Router(config)# crypto pki token default secondary config CONFIG1.CFG
```
### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>crypto pki token login</code></td>
<td>Logs in to the USB eToken.</td>
</tr>
<tr>
<td><code>crypto pki token user-pin</code></td>
<td>Creates a PIN that automatically allows the router to log into the USB eToken at router startup.</td>
</tr>
</tbody>
</table>
crypto pki token secondary unconfig

To specify a secondary “unconfig” file and its location for a USB token, use the `crypto pki token secondary unconfig` command in global configuration mode. To remove secondary configuration elements from the running configuration, use the `no` form of this command.

```
crypto pki token {token-name | default} secondary unconfig [file]

no crypto pki token {token-name | default} secondary unconfig [file]
```

**Syntax Description**

- `token-name`: Name of the token that is to be unlocked.
- `default`: Configures default values for tokens.
- `file`: (Optional) Name and location of the secondary configuration file.

**Command Default**

Secondary “unconfig” file will not be processed.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified earlier than Cisco IOS Release 15.0(1)M. The <code>default</code> keyword was added.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

Configuration files that exist on a USB token are called secondary configuration files. If you create and configure a secondary configuration file, it is executed after the token is logged in. The existence of a secondary configuration file is determined by the presence of a secondary configuration file option in the Cisco IOS configuration stored in NVRAM.

When the token is removed, logged out, or the removal timer (if set) expires, a separate “unconfig” file is processed to remove all secondary configuration elements from the running configuration. Secondary configuration and secondary “unconfig” files are executed at privilege level 15 and are not dependent on the level of the user logged in.

**Examples**

The following example shows how a secondary “unconfig” file might be used to remove secondary configuration elements from the running config. For example, a secondary configuration file might be used to set up a public key infrastructure (PKI) trustpoint. A corresponding “unconfig” file, named `mysecondaryunconfigfile.cfg`, might contain the following command:

```
no crypto pki trustpoint token-tp
```

If the token were removed and the following commands executed, the trustpoint and associated certificates would be removed from the router’s running configuration:
Router# configure terminal
Router(config)# no crypto pki token mytoken secondary unconfig mysecondaryunconfigfile.cfg

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>crypto pki token</td>
<td>Merges a specified secondary configuration file with the running configuration after the USB token is logged in to the router.</td>
</tr>
<tr>
<td></td>
<td>secondary config</td>
<td></td>
</tr>
<tr>
<td></td>
<td>crypto pki token user-pin</td>
<td>Creates a PIN that automatically allows the router to log in to the USB token at router startup.</td>
</tr>
</tbody>
</table>
crypto pki token unlock

To unlock the token and decrypt the PIN that is stored in private NVRAM, use the crypto pki token unlock command in privileged EXEC mode.

crypto pki token token-name unlock [user-pin] [passphrase passphrase]

Syntax Description

<table>
<thead>
<tr>
<th>token-name</th>
<th>Name of the token that is to be unlocked.</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-pin</td>
<td>(Optional) Specifies the USB token PIN if set.</td>
</tr>
<tr>
<td>passphrase passphrase</td>
<td>(Optional) Enables the noninteractive command-line interface (CLI). If you do not issue this keyword, you will automatically be prompted for the passphrase.</td>
</tr>
</tbody>
</table>

Tip: The noninteractive CLI is provided for instances where users will not be responding to prompts, for example in scripts, configuration tools, or other automated processes.

Note: If you are issuing this command from the console, it is recommended that you use the interactive CLI to help protect against observation from unauthorized persons.

Command Default

USB token is not unlocked, or decrypted.

Command Modes

Privileged EXEC

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(4)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
</tbody>
</table>

Usage Guidelines

After you unlock a token via the crypto pki token unlock command, the Cisco IOS software will treat the token as if it is automatically logged into the router. Any Rivest, Shamir, and Adelman (RSA) keys on the token are loaded onto the router and the secondary configuration file on the token is executed (if a secondary configuration file has been configured by the user). Secondary configuration files are executed with full user privileges.

Examples

The following example shows the configuration and encryption of a user PIN and then that the router is reloading and the user PIN is being unlocked.

```plaintext
! Configuring the user PIN

Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# crypto pki token usbtoken0: user-pin
Enter password:
```
Encrypt the user PIN

Router (config)# crypto pki token usbtoken0: encrypted-user-pin
    Enter passphrase:
Router(config)# exit
Router#

Router# show running-config

Router# crypto pki token usbtoken0 user-pin *encrypted*

Router# show running-config

Router> enable
    Password:

Router> crypto pki token usbtoken0: unlock
    Token eToken is usbtoken0

Router> crypto pki token usbtoken0: unlock

Router> show running-config

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki token</td>
<td>Creates a PIN that automatically allows the router to log into the USB</td>
</tr>
<tr>
<td>user-pin</td>
<td>token at router startup.</td>
</tr>
</tbody>
</table>
crypto pki token user-pin

To create a PIN that automatically allows the router to log in to the USB eToken at router startup, use the `crypto pki token user-pin` command in global configuration mode. To remove the stored PIN from the configuration, use the `no` form of this command.

```
crypto pki token {token-name | default} user-pin [pin] [token-pin]
```

```
no crypto pki token {token-name | default} user-pin [pin] [token-pin]
```

**Syntax Description**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>token-name</code></td>
<td>Name of USB eToken that the router will log in to.</td>
</tr>
<tr>
<td><code>default</code></td>
<td>Sets the default values for tokens.</td>
</tr>
<tr>
<td><code>user-pin</code></td>
<td>Specifies the PIN to access token.</td>
</tr>
<tr>
<td><code>pin</code></td>
<td>(Optional) User PIN required to log in to the eToken.</td>
</tr>
<tr>
<td><code>token-pin</code></td>
<td>(Optional) Token PIN name.</td>
</tr>
</tbody>
</table>

**Command Default**

If this command is not issued, the router cannot access the eToken.

**Command Modes**

Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(14)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.4(11)T</td>
<td>This command was integrated into the Cisco 7200VXR NPE-G2 platform.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified earlier than Cisco IOS Release 15.0(1)M. The <code>default</code> keyword was added.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

After the eToken is plugged into the router, the router will use the specified PIN (or the default PIN if no PIN is specified) to automatically log in as the user.

**Examples**

The following example shows how to access the eToken via the user PIN “12345”:

```
crypto pki token usbtoken0 user-pin 12345
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki login</td>
<td>Logs in to the USB eToken.</td>
</tr>
<tr>
<td>crypto pki token logout</td>
<td>Logs the router out of the USB eToken.</td>
</tr>
</tbody>
</table>
To declare the trustpoint that your router should use, use the `crypto pki trustpoint` command in global configuration mode. To delete all identity information and certificates associated with the trustpoint, use the `no` form of this command.

```
crypto pki trustpoint name redundancy
no crypto pki trustpoint name
```

### Syntax Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name</code></td>
<td>Creates a name for the trustpoint. (If you previously declared the trustpoint and just want to update its characteristics, specify the name you previously created.)</td>
</tr>
<tr>
<td><code>redundancy</code></td>
<td>(Optional) Specifies that the key, and any certificates associated with it, should be synchronized to the standby certificate authority (CA).</td>
</tr>
</tbody>
</table>

### Defaults

Your router does not recognize any trustpoints until you declare a trustpoint using this command. Your router uses unique identifiers during communication with Online Certificate Status Protocol (OCSP) servers, as configured in your network.

### Command Modes

Global configuration

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.2(8)T</td>
<td>The <code>crypto ca trustpoint</code> command was added.</td>
</tr>
<tr>
<td>12.2(15)T</td>
<td>The <code>match certificate</code> subcommand was introduced.</td>
</tr>
<tr>
<td>12.3(7)T</td>
<td>This command replaced the <code>crypto ca trustpoint</code> command. You can still enter the <code>crypto ca trusted-root</code> or <code>crypto ca trustpoint</code> command, but the command will be written in the configuration as “crypto pki trustpoint.”</td>
</tr>
<tr>
<td>12.2(18)SXD</td>
<td>This command was integrated into Cisco IOS Release 12.2(18)SXD.</td>
</tr>
<tr>
<td>12.3(14)T</td>
<td>The <code>enrollment selfsigned</code> subcommand was introduced.</td>
</tr>
<tr>
<td>12.4(4)T</td>
<td>The <code>ocsp disable-nonce</code> subcommand was introduced.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>15.0(1)M</td>
<td>This command was modified. The <code>redundancy</code> keyword was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

#### Declaring Trustpoints

Use the `crypto pki trustpoint` command to declare a trustpoint, which can be a self-signed root certificate authority (CA) or a subordinate CA. Issuing the `crypto pki trustpoint` command puts you in ca-trustpoint configuration mode.

You can specify characteristics for the trustpoint using the following subcommands:

- `crl`—Queries the certificate revocation list (CRL) to ensure that the certificate of the peer has not been revoked.
• **default (ca-trustpoint)** — Resets the value of ca-trustpoint configuration mode subcommands to their defaults.

• **enrollment** — Specifies enrollment parameters (optional).

• **enrollment http-proxy** — Accesses the CA by HTTP through the proxy server.

• **enrollment selfsigned** — Specifies self-signed enrollment (optional).

• **match certificate** — Associates a certificate-based access control list (ACL) defined with the **crypto ca certificate map** command.

• **ocsp disablenonce** — Specifies that your router will not send unique identifiers, or nonces, during OCSP communications

• **primary** — Assigns a specified trustpoint as the primary trustpoint of the router.

• **root** — Defines the TFTP to get the CA certificate and specifies both a name for the server and a name for the file that will store the CA certificate.

**Specifying Use of Unique Identifiers**

When using OCSP as your revocation method, unique identifiers, or nonces, are sent by default during peer communications with the OCSP server. The use of unique identifiers during OCSP server communications enables more secure and reliable communications. However, not all OCSP servers support the use of unique identifiers, see your OCSP manual for more information. To disable the use of unique identifiers during OCSP communications, use the **ocsp disable-nonc**e subcommand.

**Examples**

The following example shows how to declare the CA named *ka* and specify enrollment and CRL parameters:

```
crypto pki trustpoint ka
  enrollment url http://kahului:80
```

The following example shows a certificate-based ACL with the label Group defined in a **crypto pki certificate map** command and included in the **match certificate** subcommand of the **crypto pki trustpoint** command:

```
crypto pki certificate map Group 10
  subject-name co=wan
  subject-name co=cisco
!
crypto pki trustpoint pki1
  match certificate Group
```

The following example shows a self-signed certificate being designated for a trustpoint named local using the **enrollment selfsigned** subcommand of the **crypto pki trustpoint** command:

```
crypto pki trustpoint local
  enrollment selfsigned
```

The following example shows the unique identifier being disabled for OCSP communications for a previously created trustpoint named *ts*:

```
crypto pki trustpoint ts
  ocsp disable-nonc
```

The following example shows the **redundancy** keyword specified in the **crypto pki trustpoint** command:

```
Router(config)#crypto pki trustpoint mytp
Router(ca-trustpoint)#redundancy
Router(ca-trustpoint)#show
```
Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crl</td>
<td>Queries the CRL to ensure that the certificate of the peer has not been revoked.</td>
</tr>
<tr>
<td>default (ca-trustpoint)</td>
<td>Resets the value of a ca-trustpoint configuration subcommand to its default.</td>
</tr>
<tr>
<td>enrollment</td>
<td>Specifies the enrollment parameters of your CA.</td>
</tr>
<tr>
<td>enrollment http-proxy</td>
<td>Accesses the CA by HTTP through the proxy server.</td>
</tr>
<tr>
<td>primary</td>
<td>Assigns a specified trustpoint as the primary trustpoint of the router.</td>
</tr>
<tr>
<td>root</td>
<td>Obtains the CA certificate via TFTP.</td>
</tr>
</tbody>
</table>
crypto provisioning petitioner

To configure a device to become an easy secure device provisioning (SDP) petitioner and enter tti-petitioner configuration mode, use the `crypto provisioning petitioner` command in global configuration mode. To disable petitioner support, use the `no` form of this command.

```
crypto provisioning petitioner
no crypto provisioning petitioner
```

**Syntax Description**

This command has no arguments or keywords.

**Defaults**

A device (with a crypto image) is configured to be an SDP petitioner.

**Command Modes**

Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2.3(8)T</td>
<td>The <code>crypto wui tti petitioner</code> command was introduced.</td>
</tr>
<tr>
<td>T2.3(14)T</td>
<td>This command replaced the <code>crypto wui tti petitioner</code> command.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**

SDP uses Trusted Transitive Introduction (TTI) to easily deploy public key infrastructure (PKI) between two end devices. TTI, which is a communication protocol that provides a bidirectional introduction between two end entities, involves the following three entities:

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

**Note**

Because the petitioner is enabled by default on the device, you only have to issue the `crypto provisioning petitioner` command if you have previously disabled the petitioner or if you want to use an existing trustpoint instead of the automatically generated trustpoint.

**Examples**

After the SDP exchange is complete, the petitioner will automatically enroll with the registrar and obtain a certificate. The following sample output from the `show running-config` command shows an automatically generated configuration at the petitioner.

```
Note

The petitioner will not have any TTI-specific configuration in the beginning except that the IP HTTP server will be turned on and the Domain Name System (DNS) server needs to be properly configured.)
```
crypto pki trustpoint tti
! Enrollment url contains the registrar CS details
enrollment url http://pki1-36a.cisco.com:80
revocation-check crl
rsakeypair tti 1024
auto-enroll 70

<table>
<thead>
<tr>
<th>Related Commands</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>crypto provisioning registrar</td>
<td>Configures a device to become an SDP registrar and enters tti-registrar configuration mode.</td>
</tr>
<tr>
<td></td>
<td>trustpoint (tti-petitioner)</td>
<td>Specifies the trustpoint that is to be associated with the TTI exchange between the SDP petitioner and the SDP registrar.</td>
</tr>
</tbody>
</table>
crypto provisioning registrar

To configure a device to become an easy secure device provisioning (SDP) registrar and enter tti-registrar configuration mode, use the **crypto provisioning registrar** command in global configuration mode. To disable registrar support, use the **no** form of this command.

**crypto provisioning registrar**

**no crypto provisioning registrar**

**Syntax Description**
This command has no arguments or keywords.

**Defaults**
The registrar is not enabled.

**Command Modes**
Global configuration

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(8)T</td>
<td>The <strong>crypto wui tti registrar</strong> command was introduced.</td>
</tr>
<tr>
<td>12.3(14)T</td>
<td>This command replaced the <strong>crypto wui tti registrar</strong> command.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**
SDP uses Trusted Transitive Introduction (TTI) to easily deploy public key infrastructure (PKI) between two end devices. TTI, which is a communication protocol that provides a bidirectional introduction between two end entities, involves the following three entities:

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

Although any device that contains a crypto image can be the registrar, it is recommended that the registrar be either a Cisco IOS certificate server registration authority (RA) or a Cisco IOS certificate server root.

**Examples**
The following sample output from the **show running-config** command verifies that the certificate server “cs1” was configured and associated with the TTI exchange between the registrar and petitioner:

crypto pki server cs1
  issuer-name CN = ioscs,L = Santa Cruz,C =US
  lifetime crl 336
  lifetime certificate 730
  !
crypto pki trustpoint pki-36a
  enrollment url http://pki-36a:80
  ip-address FastEthernet0/0
  revocation-check none
  !
crypto provisioning registrar
quit
!
crypto provisioning registrar
pki-server cs1
!
!
crypto isakmp policy 1
  hash md5
  
!
crypto ipsec transform-set test_transformset esp-3des
!
crypto map test_cryptomap 10 ipsec-isakmp
  set peer 10.23.1.10
  set security-association lifetime seconds 1800
  set transform-set test_transformset
  match address 170

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server</td>
</tr>
<tr>
<td></td>
<td>configuration mode.</td>
</tr>
<tr>
<td>crypto provisioning</td>
<td>Configures a device to become an SDP petitioner and enters tti-petitioner</td>
</tr>
<tr>
<td>petitioner</td>
<td>configuration mode.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
crypto wui tti petitioner

Note

This command was replaced by the `crypto provisioning petitioner` command effective with Cisco IOS Release 12.3(14)T.

To configure a device to become an easy secure device deployment (EzSDD) petitioner and enter tti-petitioner configuration mode, use the `crypto wui tti petitioner` command in global configuration mode. To disable petitioner support, use the `no` form of this command.

```
crypto wui tti petitioner

no crypto wui tti petitioner
```

Syntax Description

This command has no arguments or keywords.

Defaults

A device (with a crypto image) is configured to be an EzSDD petitioner.

Command Modes

Global configuration

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(8)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

Usage Guidelines

EzSDD uses Trusted Transitive Introduction (TTI) to easily deploy public key infrastructure (PKI) between two end devices. TTI, which is a communication protocol that provides a bidirectional introduction between two end entities, involves the following three entities:

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

Note

Because the petitioner is enabled by default on the device, you only have to issue the `crypto wui tti petitioner` command if you have previously disabled the petitioner or if you want to use an existing trustpoint instead of the automatically generated trustpoint.
Examples
After the EzSDD exchange is complete, the petitioner will automatically enroll with the registrar and obtain a certificate. The following sample output from the `show running-config` command shows an automatically generated configuration at the petitioner. (Note that petitioner will not have any TTI-specific configuration in the beginning except that the http server will be turned on and the Domain Name System (DNS) server needs to be properly configured.)

```
crypto pki trustpoint tti
! Enrollment url contains the registrar CS details
enrollment url http://pki1-36a.cisco.com:80
revocation-check crl
rsakeypair tti 1024
auto-enroll 70
```

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto wui tti registrar</td>
<td>Configures a device to become an EzSDD registrar and enters tti-registrar configuration mode.</td>
</tr>
<tr>
<td>trustpoint (tti-petitioner)</td>
<td>Specifies the trustpoint that is to be associated with the TTI exchange between the EzSDD petitioner and the EzSDD registrar.</td>
</tr>
</tbody>
</table>
crypto wui tti registrar

Note
This command was replaced by the crypto provisioning registrar command effective with Cisco IOS Release 12.3(14)T.

To configure a device to become an easy secure device deployment (EzSDD) registrar and enter tti-registrar configuration mode, use the crypto wui tti registrar command in global configuration mode. To disable registrar support, use the no form of this command.

```
crypto wui tti registrar

no crypto wui tti registrar
```

Syntax Description
This command has no arguments or keywords.

Defaults
The registrar is not enabled.

Command Modes
Global configuration

Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.3(8)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

Usage Guidelines
EzSDD uses Trusted Transitive Introduction (TTI) to easily deploy public key infrastructure (PKI) between two end devices. TTI, which is a communication protocol that provides a bidirectional introduction between two end entities, involves the following three entities:

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

Although any device that contains a crypto image can be the registrar, it is recommended that the registrar be either a Cisco IOS certificate server registration authority (RA) or a Cisco IOS certificate server root.

Examples
The following sample output from the show running-config command verifies that the certificate server “cs1” was configured and associated with the TTI exchange between the registrar and petitioner:

```
crypto pki server cs1
  issuer-name CN = ioscsl, L = Santa Cruz, C =US
  lifetime crl 336
  lifetime certificate 730
```

```
crypto pki trustpoint pki-36a
```
Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto pki server</td>
<td>Enables a Cisco IOS certificate server and enters certificate server configuration mode.</td>
</tr>
<tr>
<td>crypto wui tti petitioner</td>
<td>Configures a device to become an EzSDD petitioner and enters tti-petitioner configuration mode.</td>
</tr>
</tbody>
</table>
crypto xauth

To configure crypto Extended Authentication (xauth) parameters globally on a per-interface basis, use the **crypto xauth** command in global configuration mode. To disable the xauth parameters, use the **no** form of this command.

```
crypto xauth interface-name interface-number
no crypto xauth interface-name interface-number
```

<table>
<thead>
<tr>
<th>Syntax Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface-name</td>
<td>Name of the interface.</td>
</tr>
<tr>
<td>interface-number</td>
<td>Number of the related interface. Each interface has a related range of numbers. For example, the asynchronous interface has a range of interface numbers from 1 to 5 and the BVI interface has a range of interface numbers from 1 to 255.</td>
</tr>
</tbody>
</table>

**Command Default**
Crypto xauth parameters are not configured on any interface.

**Command Modes**
Global configuration (config)

**Command History**

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.0(1)M</td>
<td>This command was introduced in a release earlier than Cisco IOS 15.0(1)M.</td>
</tr>
</tbody>
</table>

**Usage Guidelines**
This command is mainly used on responders.

This command is used to disable the negotiation of xauth capabilities during proposals for a session that is terminating on a specific interface.

The **no crypto xauth** command enables the negotiation of xauth capabilities.

**Examples**
The following example shows how to enable crypto xauth parameters globally on a per-interface basis:

```
Router> enable
Router# configure terminal
Router(config)# crypto xauth fastethernet 0/1
```

The following example shows how the **no crypto xauth** command uses the nonvolatile generation (NVGEN) process to perform a configuration state retrieval operation when you specify the **show run** command:

```
Router> enable
Router# configure terminal
Router(config)# no crypto xauth fastethernet 0/1
```

```
Router# show run
archive
log config
```
hidekeys
!
redundancy
!
!
no crypto xauth Ethernet0/0

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto key decrypt rsa</td>
<td>Deletes the encrypted RSA key and leaves only the unencrypted key on the running router.</td>
</tr>
</tbody>
</table>
To enable Cisco Secure Desktop (CSD) support for SSL VPN sessions, use the `csd enable` command in webvpn context configuration mode. To remove CSD support from the SSL VPN context configuration, use the `no csd enable` form of this command.

```plaintext
  csd enable

  no csd enable
```

**Syntax Description**

This command has no keywords or arguments.

**Command Default**

CSD support is not enabled.

**Command Modes**

Webvpn context configuration

**Command History**

 Release | Modification 
---------- | -------------- 
 T2.4(6)T | This command was introduced. 

**Usage Guidelines**

The CSD software installation package must be present in a local file system, such as flash memory, and it must be cached for distribution to end users (remote PC or networking device). The `webvpn install` command is used to install the software installation package to the distribution cache.

**Examples**

The following example enables CSD support for SSL VPN sessions:

```plaintext
Router(config)# webvpn install csd flash:/securedesktop_3_1_0_9.pkg
SSLVPN Package Cisco-Secure-Desktop : installed successfully
Router(config)# webvpn context context1
Router(config-webvpn-context)# csd enable
```

**Related Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>webvpn context</td>
<td>Enters webvpn context configuration mode to configure the SSL VPN context.</td>
</tr>
<tr>
<td>webvpn install</td>
<td>Installs a CSD or SSL VPN client package file to a SSL VPN gateway for distribution to end users.</td>
</tr>
</tbody>
</table>
ctcp port

To set the port number for Cisco Tunneling Control Protocol (cTCP) encapsulation for Easy VPN, use the `ctcp port` command in crypto ipsec client ezvpn configuration mode. To disable the port that was configured, use the `no` form of this command.

```
ctcp port port-number
```

```
no ctcp port
```

### Syntax Description

| port-number | Port number. Value = 1 through 65535. |

### Command Default

If a port is not specified, the default port is the port on which the cTCP server listens.

### Command Modes

Crypto ipsec client ezvpn configuration (config-crypto-ezvpn)

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.4(20)T</td>
<td>This command was introduced.</td>
</tr>
</tbody>
</table>

### Usage Guidelines

This command is used only on the Easy VPN remote device.

### Examples

The following example shows that the cTCP port number has been set to 10:

```
Router (config)# crypto ipsec client ezvpn client1
Router (config-crypto-ezvpn)# ctcp port 10
```

### Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crypto ctcp</td>
<td>Configures cTCP encapsulation for Easy VPN.</td>
</tr>
</tbody>
</table>
To preauthenticate calls on the basis of the call type, use the `ctype` command in AAA preauthentication configuration mode. To remove the `ctype` command from your configuration, use the `no` form of this command.

```
ctype [if_avail | required] [accept-stop] [password password] [digital | speech | v.110 | v.120]
```

```
no ctype [if_avail | required] [accept-stop] [password password] [digital | speech | v.110 | v.120]
```

### Syntax Description

- `if_avail` (Optional) Implies that if the switch provides the data, RADIUS must be reachable and must accept the string in order for preauthentication to pass. If the switch does not provide the data, preauthentication passes.

- `required` (Optional) Implies that the switch must provide the associated data, that RADIUS must be reachable, and that RADIUS must accept the string in order for preauthentication to pass. If these three conditions are not met, preauthentication fails.

- `accept-stop` (Optional) Prevents subsequent preauthentication elements such as `clid` or `dnis` from being tried once preauthentication has succeeded for a call element.

- `password password` (Optional) Defines the password for the preauthentication element.

- `digital` (Optional) Specifies “digital” as the call type for preauthentication.

- `speech` (Optional) Specifies “speech” as the call type for preauthentication.

- `v.110` (Optional) Specifies “v.110” as the call type for preauthentication.

- `v.120` (Optional) Specifies “v.120” as the call type for preauthentication.

### Defaults

The `if_avail` and `required` keywords are mutually exclusive. If the `if_avail` keyword is not configured, the preauthentication setting defaults to `required`. The default password string is cisco.

### Command Modes

AAA preauthentication configuration

### Command History

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1(2)T</td>
<td>This command was introduced.</td>
</tr>
<tr>
<td>12.2(33)SRA</td>
<td>This command was integrated into Cisco IOS Release 12.2(33)SRA.</td>
</tr>
<tr>
<td>12.2SX</td>
<td>This command is supported in the Cisco IOS Release 12.2SX train. Support in a specific 12.2SX release of this train depends on your feature set, platform, and platform hardware.</td>
</tr>
</tbody>
</table>
Usage Guidelines

You may configure more than one of the AAA preauthentication commands (clid, ctype, dnis) to set conditions for preauthentication. The sequence of the command configuration decides the sequence of the preauthentication conditions. For example, if you configure dnis, then clid, then ctype, in this order, then this is the order of the conditions considered in the preauthentication process.

In addition to using the preauthentication commands to configure preauthentication on the Cisco router, you must set up the preauthentication profiles on the RADIUS server.

Set up the RADIUS preauthentication profile with the call type string as the username and with the password that is defined in the ctype command as the password. Table 28 shows the call types that you may use in the preauthentication profile.

Table 28  Preauthentication Call Types

<table>
<thead>
<tr>
<th>Call Type String</th>
<th>ISDN Bearer Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>digital</td>
<td>Unrestricted digital, restricted digital.</td>
</tr>
<tr>
<td>speech</td>
<td>Speech, 3.1 kHz audio, 7 kHz audio.</td>
</tr>
<tr>
<td>v.110</td>
<td>Anything with V.110 user information layer.</td>
</tr>
<tr>
<td>v.120</td>
<td>Anything with V.120 user information layer.</td>
</tr>
</tbody>
</table>

Examples

The following example specifies that incoming calls be preauthenticated on the basis of the call type:

```
aaa preauth
  group radius
  ctype required
```

Related Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clid</td>
<td>Preauthenticates calls on the basis of the CLID number.</td>
</tr>
<tr>
<td>dnis (RADIUS)</td>
<td>Preauthenticates calls on the basis of the DNIS number.</td>
</tr>
<tr>
<td>dnis bypass (AAA preauthentication configuration)</td>
<td>Specifies a group of DNIS numbers that will be bypassed for preauthentication.</td>
</tr>
<tr>
<td>group (RADIUS)</td>
<td>Specifies the AAA RADIUS server group to use for preauthentication.</td>
</tr>
</tbody>
</table>