Configuring Certification Authority Interoperability

Feature History

Release	Modification
Cisco IOS	For information about feature support in Cisco IOS software, use Cisco Feature Navigator.

This chapter describes how to configure certification authority (CA) interoperability, which is provided in support of the IP Security (IPSec) protocol. CA interoperability permits Cisco IOS devices and CAs to communicate so that your Cisco IOS device can obtain and use digital certificates from the CA. Although IPSec can be implemented in your network without the use of a CA, using a CA provides manageability and scalability for IPSec.

For background and configuration information for IPSec, see the chapter "Configuring IPSec Network Security."

For a complete description of the commands used in this chapter, refer to the chapter "Certification Authority Interoperability Commands" in the *Cisco IOS Security Command Reference*. To locate documentation for other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Supported Platforms" section in the chapter "Using Cisco IOS Software."

In This Chapter

This chapter contains the following sections:

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

About CA Interoperability

Without CA interoperability, Cisco IOS devices could not use CAs when deploying IPSec. CAs provide a manageable, scalable solution for IPSec networks. For details, see the section "About Certification Authorities."

This section contains the following sections:

Supported Standards

Cisco supports the following standards with this feature:

- IPSec—IP Security Protocol. IPSec is a framework of open standards that provides data confidentiality, data integrity, and data authentication between participating peers. IPSec provides these security services at the IP layer; it uses Internet Key Exchange to handle negotiation of protocols and algorithms based on local policy, and to generate the encryption and authentication keys to be used by IPSec. IPSec can be used to protect one or more data flows between a pair of hosts, between a pair of security gateways, or between a security gateway and a host.

  For more information on IPSec, see the chapter "Configuring IPSec Network Security."

- Internet Key Exchange (IKE)—A hybrid protocol that implements Oakley and Skeme key exchanges inside the Internet Security Association Key Management Protocol (ISAKMP) framework. Although IKE can be used with other protocols, its initial implementation is with the IPSec protocol. IKE provides authentication of the IPSec peers, negotiates IPSec keys, and negotiates IPSec security associations.

  For more information on IKE, see the chapter "Configuring Internet Key Exchange Security Protocol."

- Public-Key Cryptography Standard #7 (PKCS #7)—A standard from RSA Data Security, Inc., used to encrypt and sign certificate enrollment messages.
- Public-Key Cryptography Standard #10 (PKCS #10)—A standard syntax from RSA Data Security, Inc. for certificate requests.
- RSA Keys—RSA is the public key cryptographic system developed by Ron Rivest, Adi Shamir, and Leonard Adleman. RSA keys come in pairs: one public key and one private key.
- X.509v3 certificates—Certificate support that allows the IPSec-protected network to scale by providing the equivalent of a digital ID card to each device. When two devices wish to communicate, they exchange digital certificates to prove their identity (thus removing the need to manually exchange public keys with each peer or to manually specify a shared key at each peer). These certificates are obtained from a certification authority (CA). X.509 is part of the X.500 standard of the ITU.

Restrictions

When configuring your CA, the following restrictions apply:

- This feature should be configured only when you also configure both IPSec and IKE in your network.
- The Cisco IOS software *does not* support CA server public keys greater than 2048 bits.

Prerequisites

You need to have a certification authority (CA) available to your network before you configure this interoperability feature. The CA must support Cisco Systems' PKI protocol, the Simple Certificate Enrollment Protocol (SCEP) (formerly called certificate enrollment protocol (CEP)).

About Certification Authorities

This section provides background information about CAs, including the following:

Purpose of CAs

CAs are responsible for managing certificate requests and issuing certificates to participating IPSec network devices. These services provide centralized key management for the participating devices.

CAs simplify the administration of IPSec network devices. You can use a CA with a network containing multiple IPSec-compliant devices such as routers.

Digital signatures, enabled by public key cryptography, provide a means of digitally authenticating devices and individual users. In public key cryptography, such as the RSA encryption system, each user has a key pair containing both a public and a private key. The keys act as complements, and anything encrypted with one of the keys can be decrypted with the other. In simple terms, a signature is formed when data is encrypted with a user's private key. The receiver verifies the signature by decrypting the message with the sender's public key. The fact that the message could be decrypted using the sender's public key indicates that the holder of the private key, the sender, must have created the message. This process relies on the receiver's having a copy of the sender's public key and knowing with a high degree of certainty that it really does belong to the sender and not to someone pretending to be the sender.

Digital certificates provide the link. A digital certificate contains information to identify a user or device, such as the name, serial number, company, department, or IP address. It also contains a copy of the entity's public key. The certificate is itself signed by a certification authority (CA), a third party that is explicitly trusted by the receiver to validate identities and to create digital certificates.

In order to validate the signature of the CA, the receiver must first know the CA's public key. Normally this process is handled out-of-band or through an operation done at installation. For instance, most web browsers are configured with the public keys of several CAs by default. The Internet Key Exchange (IKE), an essential component of IPSec, can use digital signatures to scalably authenticate peer devices before setting up security associations.

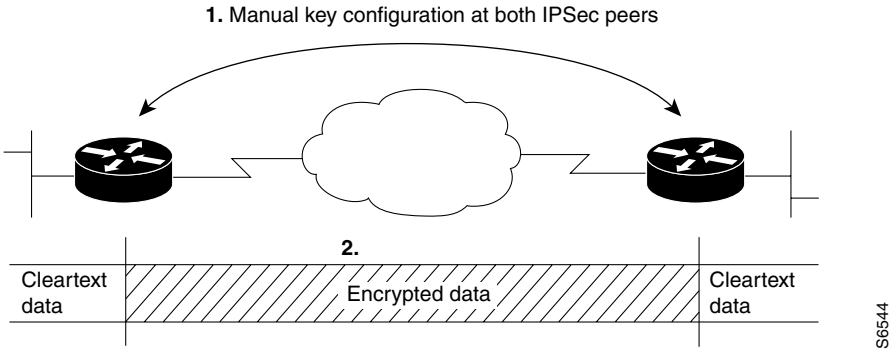
Without digital signatures, one must manually exchange either public keys or secrets between each pair of devices that use IPSec to protect communications between them. Without certificates, every new device added to the network requires a configuration change on every other device with which it communicates securely. With digital certificates, each device is enrolled with a certification authority. When two devices wish to communicate, they exchange certificates and digitally sign data to authenticate each other. When a new device is added to the network, one simply enrolls that device with a CA, and none of the other devices needs modification. When the new device attempts an IPSec connection, certificates are automatically exchanged and the device can be authenticated.

Implementing IPSec Without CAs

Without a CA, if you want to enable IPSec services (such as encryption) between two Cisco routers, you must first ensure that each router has the key of the other router (such as an RSA public key or a shared key). This requirement means that you must manually perform one of the following operations:

- At each router, enter the RSA public key of the other router
- At each router, specify a shared key to be used by both routers

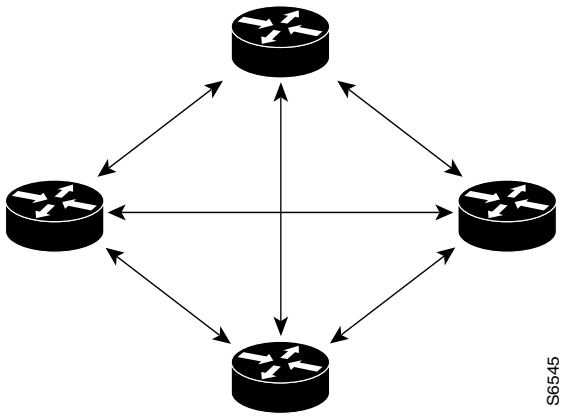
Figure 33 Without a CA: Key Configuration Between Two Routers



In Figure 33, each router uses the key of the other router to authenticate the identity of the other router; this authentication always occurs when IPSec traffic is exchanged between the two routers.

If you have multiple Cisco routers in a mesh topology and wish to exchange IPSec traffic passing among all of those routers, you must first configure shared keys or RSA public keys among all of those routers.

Figure 34 Without a CA: Six Two-Part Key Configurations Required for Four IPSec Routers



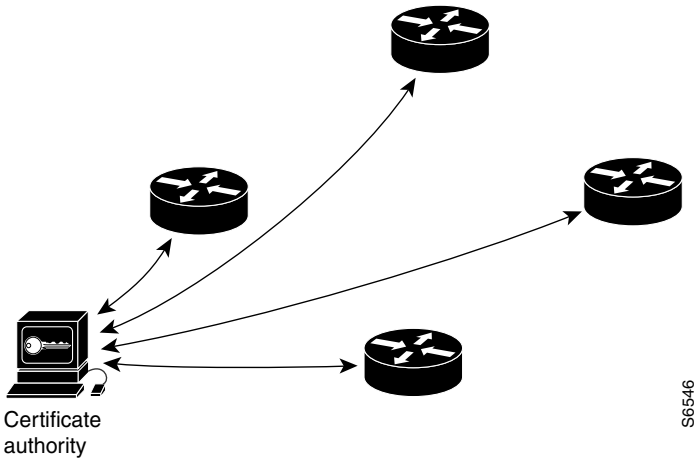
Every time a new router is added to the IPSec network, you must configure keys between the new router and each of the existing routers. (In Figure 34, four additional two-part key configurations would be required to add a single encrypting router to the network.)

Consequently, the more devices there are that require IPSec services, the more involved the key administration becomes. This approach does not scale well for larger, more complex encrypting networks.

Implementing IPSec with CAs

With a CA, you do not have to configure keys between all the encrypting routers. Instead, you individually enroll each participating router with the CA, requesting a certificate for the router. When this has been accomplished, each participating router can dynamically authenticate all the other participating routers. This process is illustrated in Figure 35.

Figure 35 With a CA: Each Router Individually Makes Requests of the CA at Installation



To add a new IPSec router to the network, you need only configure that new router to request a certificate from the CA, instead of making multiple key configurations with all the other existing IPSec routers.

Implementing IPSec with Multiple Root CAs

With multiple root CAs, you no longer have to enroll a router with the CA that issued a certificate to a peer. Instead, you configure a router with multiple CAs that it trusts. Thus, a router can use a configured CA (a trusted root) to verify certificates offered by a peer that were not issued by the same CA defined in the identity of the router.

Configuring multiple CAs allows two or more routers enrolled under different domains (different CAs) to verify the identity of each other when using IKE to set up IPSec tunnels.

Through Simple Certificate Enrollment Protocol (SCEP), each router is configured with a CA (the enrollment CA). The CA issues a certificate to the router that is signed with the private key of the CA. To verify the certificates of peers in the same domain, the router is also configured with the root certificate of the enrollment CA.

To verify the certificate of a peer from a different domain, the root certificate of the enrollment CA in the domain of the peer must be configured securely in the router.

During IKE phase one signature verification, the initiator will send the responder a list of its CA certificates. The responder should send the certificate issued by one of the CAs in the list. If the certificate is verified, the router saves the public key contained in the certificate on its public key ring.

With multiple root CAs, Virtual Private Network (VPN) users can establish trust in one domain and easily and securely distribute it to other domains. Thus, the required private communication channel between entities authenticated under different domains can occur.

How CA Certificates Are Used by IPSec Devices

When two IPSec routers want to exchange IPSec-protected traffic passing between them, they must first authenticate each other—otherwise, IPSec protection cannot occur. The authentication is done with IKE.

Without a CA, a router authenticates itself to the remote router using either RSA-encrypted nonces or preshared keys. Both methods require that keys must have been previously configured between the two routers.

With a CA, a router authenticates itself to the remote router by sending a certificate to the remote router and performing some public key cryptography. Each router must send its own unique certificate that was issued and validated by the CA. This process works because the certificate of each router encapsulates the public key of the router, each certificate is authenticated by the CA, and all participating routers recognize the CA as an authenticating authority. This scheme is called IKE with an RSA signature.

Your router can continue sending its own certificate for multiple IPSec sessions, and to multiple IPSec peers until the certificate expires. When its certificate expires, the router administrator must obtain a new one from the CA.

CAs can also revoke certificates for devices that will no longer participate in IPSec. Revoked certificates are not recognized as valid by other IPSec devices. Revoked certificates are listed in a certificate revocation list (CRL), which each peer may check before accepting a certificate from another peer.

About Registration Authorities

Some CAs have a registration authority (RA) as part of their implementation. An RA is essentially a server that acts as a proxy for the CA so that CA functions can continue when the CA is offline.

Some of the configuration tasks described in this document differ slightly, depending on whether your CA supports an RA.

CA Interoperability Configuration Task Lists

To enable your Cisco device to interoperate with a CA, complete the tasks in the following sections. Some of the tasks are optional; the remaining are required.

- Managing NVRAM Memory Usage (Optional)
- Configuring the Routers Host Name and IP Domain Name (Required)
- Generating an RSA Key Pair (Required)
- Declaring a Certification Authority (Required)
- Configuring a Root CA (Trusted Root) (Optional)
- Authenticating the CA (Required)
- Requesting Your Own Certificates (Required)
- Saving Your Configuration (Required)
- Monitoring and Maintaining Certification Authority Interoperability (Optional)

For CA interoperability configuration examples, refer to the section "CA Interoperability Configuration Examples" at the end of this chapter.

Managing NVRAM Memory Usage

Certificates and certificate revocation lists (CRLs) are used by your router when a CA is used. Normally certain certificates and all CRLs are stored locally in the router's NVRAM, and each certificate and CRL uses a moderate amount of memory.

The following certificates are normally stored at your router:

- The certificate of your router
- The certificate of the CA
- Root certificates obtained from CA servers (all root certificates are saved in RAM after the router has been initialized)
- Two registration authority (RA) certificates (only if the CA supports an RA)

CRLs are normally stored at your router according to the following conditions:

- If your CA does not support an RA, only one CRL gets stored at your router.
- If your CA supports an RA, multiple CRLs can be stored at your router.

In some cases, storing these certificates and CRLs locally will not present any difficulty. In other cases, memory might become a problem—particularly if your CA supports an RA and a large number of CRLs have to be stored on your router. If the NVRAM is too small to store root certificates, only the fingerprint of the root certificate will be saved.

To save NVRAM space, you can specify that certificates and CRLs should not be stored locally, but should be retrieved from the CA when needed. This alternative will save NVRAM space but could result in a slight performance impact.

To specify that certificates and CRLs should not be stored locally on your router, but should be retrieved when required, turn on query mode by using the following command in global configuration mode:

Command	Purpose
`Router(config)# crypto ca certificate query`	Turns on query mode, which causes certificates and CRLs not to be stored locally.

Note Query mode may affect availability if the CA is down.

If you do not turn on query mode now, you can turn it on later even if certificates and CRLs have already been stored on your router. In this case, when you turn on query mode, the stored certificates and CRLs will be deleted from the router after you save your configuration. (If you copy your configuration to a TFTP site prior to turning on query mode, you will save any stored certificates and CRLs at the TFTP site.)

If you turn on query mode now, you can turn off query mode later if you wish. If you turn off query mode later, you could also perform the **copy system:running-config nvram:startup-config** command at that time to save all current certificates and CRLs to NVRAM. Otherwise they could be lost during a reboot and would need to be retrieved the next time they were needed by your router.

Configuring the Routers Host Name and IP Domain Name

You must configure the host name and IP domain name of the router if this has not already been done. This is required because the router assigns a fully qualified domain name (FQDN) to the keys and certificates used by IPSec, and the FQDN is based on the host name and IP domain name you assign to the router. For example, a certificate named "router20.example.com" is based on a router host name of "router20" and a router IP domain name of "example.com".

To configure the host name and IP domain name of the router, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# hostname` *name*	Configures the host name of the router.
Step 2	`Router(config)# ip domain-name` *name*	Configures the IP domain name of the router.

Generating an RSA Key Pair

RSA key pairs are used to sign and encrypt IKE key management messages and are required before you can obtain a certificate for your router.

To generate an RSA key pair, use the following command in global configuration mode:

Command	Purpose
`Router(config)# crypto key generate rsa [usage-keys]`	Generates an RSA key pair. Use the **usage-keys** keyword to specify special-usage keys instead of general-purpose keys. See the *Cisco IOS Security Command Reference* for an explanation of special-usage versus general-purpose keys for this command.

Declaring a Certification Authority

You should declare one certification authority (CA) to be used by your router.

To declare a CA, use the following commands, beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# crypto ca identity name`	Declares a CA. The name should be the domain name of the CA. This command puts you into the ca-identity configuration mode.
Step 2	`Router(ca-identity)# enrollment url url`	Specifies the URL of the CA. (The URL should include any nonstandard cgi-bin script location.)
Step 3	`Router(ca-identity)# enrollment mode ra`	(Optional) Specifies RA mode if your CA system provides a registration authority (RA). **Note** The Cisco IOS software automatically determines the mode—RA or non-RA; therefore, if RA mode is used, this subcommand is written to NVRAM during "write memory."
Step 4	`Router(ca-identity)# query url url`	Specifies the location of the LDAP server if your CA system provides an RA and supports the LDAP protocol.
Step 5	`Router(ca-identity)# enrollment retry period minutes`	(Optional) Specifies a retry period. After requesting a certificate, the router waits to receive a certificate from the CA. If the router does not receive a certificate within a period of time (the retry period) the router will send another certificate request. You can change the retry period from the default of 1 minute.
Step 6	`Router(ca-identity)# enrollment retry count number`	(Optional) Specifies how many times the router will continue to send unsuccessful certificate requests before giving up. By default, the router will never give up trying.
Step 7	`Router(ca-identity)# crl optional`	(Optional) Specifies that other peers' certificates can still be accepted by your router even if the appropriate CRL is not accessible to your router.
Step 8	`Router(ca-identity)# exit`	Exits ca-identity configuration mode.

The trade-off between security and availability is determined by the **query url** and **crl optional** commands, as shown in Table 26.

Table 26 Security and CA Availability

	Query—Yes	Query—No
CRL Optional—Yes	Sessions will go through even if the CA is not available, but the certificate may have been revoked.	Sessions will go through even if the CA is not available, but the certificate may have been revoked.
CRL Optional—No	Certificates will not be accepted if the CA is not available.	Sessions will go through, and will be verified against the CRL stored locally.

Configuring a Root CA (Trusted Root)

To configure a trusted root, enter the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# crypto ca trusted-root name`	Configures a root with a selected name and enters trusted root configuration mode.
Step 2	`Router(ca-root)# crl query url`	(Optional) Queries the CRL published by the configured root with the LDAP[1] URL.
Step 3	`Router(ca-root)# exit`	(Optional) Exits trusted root configuration mode.
Step 4	`Router(config)# crypto ca identity name`	(Optional) Enters certificate authority identity configuration mode.
Step 5	`Router(ca-identity)# crl optional`	(Optional) Allows other peer certificates to be accepted by your router even if the appropriate CRL is not accessible to your router.
Step 6	`Router(ca-identity)# exit`	(Optional) Exits certificate authority identity configuration mode.
Step 7	`Router(config)# crypto ca trusted-root name`	(Optional) Enters trusted root configuration mode.

Step 8	`Router(ca-root)# root CEP url`
	or
	`Router(ca-root)# root TFTP server-hostname filename`
		Uses SCEP[2], with the given identity and URL, to get a root certificate.
		or
		Uses TFTP to get a root certificate.
		Note Only use TFTP if your CA server does not support SCEP.
		Note When you are using TFTP, the server must be secured so that the downloading of the root certificate is not subject to any attack.

Step 9	`Router(ca-root)# root PROXY url`	Defines the HTTP proxy server for getting a root certificate.

1. LDAP = Lightweight Directory Access Protocol.
2. SCEP = Simple Certificate Ennrollment Protocol (formerly called Cisco Enrollment Protocol (CEP)).

Authenticating the CA

The router must authenticate the CA. It does this by obtaining the self-signed certificate of the CA, which contains the public key of the CA. Because the certificate of the CA is self-signed (the CA signs its own certificate) the public key of the CA should be manually authenticated by contacting the CA administrator to compare the fingerprint of the CA certificate when you perform this step.

To get the public key of the CA, use the following command in global configuration mode:

Command	Purpose
`Router(config)#` **`crypto ca authenticate`** *`name`*	Gets the public key of the CA. Use the same *name* that you used when declaring the CA or when using the **crypto ca identity** command.

Requesting Your Own Certificates

You must obtain a signed certificate from the CA for each of your router's RSA key pairs. If you generated general-purpose RSA keys, your router has only one RSA key pair and needs only one certificate. If you previously generated special-usage RSA keys, your router has two RSA key pairs and needs two certificates.

To request signed certificates from the CA, use the following command in global configuration mode:

Command	Purpose
`Router(config)#` **`crypto ca enroll`** *`name`*	Requests certificates for all of your RSA key pairs. This command causes your router to request as many certificates as there are RSA key pairs, so you need only perform this command once, even if you have special-usage RSA key pairs. **Note** This command requires you to create a challenge password that is not saved with the configuration. This password is required in the event that your certificate needs to be revoked, so you must remember this password.

Note If your router reboots after you have issued the **crypto ca enroll** command but before you have received the certificates, you must reissue the command and notify the CA administrator.

Saving Your Configuration

Always remember to save your work when you make configuration changes.

Use the **copy system:running-config nvram:startup-config** command to save your configuration. This command includes saving RSA keys to private NVRAM. RSA keys are *not* saved with your configuration when you use a **copy system:running-config rcp:** or **copy system:running-config tftp:** command.

Monitoring and Maintaining Certification Authority Interoperability

The following tasks are optional, depending on your particular requirements:

- Requesting a Certificate Revocation List
- Querying a Certificate Revocation List
- Deleting RSA Keys from Your Router

- Deleting a Peer's Public Keys
- Deleting Certificates from the Configuration
- Viewing Keys and Certificates

Requesting a Certificate Revocation List

You can request a certificate revocation list (CRL) only if your CA does not support a registration authority (RA). The following description and task applies only when the CA does not support an RA.

When your router receives a certificate from a peer, your router will download a CRL from the CA. Your router then checks the CRL to make sure the certificate that the peer sent has not been revoked. (If the certificate appears on the CRL, your router will not accept the certificate and will not authenticate the peer.)

A CRL can be reused with subsequent certificates until the CRL expires if query mode is off. If your router receives a peer's certificate after the applicable CRL has expired, the router will download the new CRL.

If your router has a CRL that has not yet expired, but you suspect that the contents of the CRL are out of date, you can request that the latest CRL be downloaded immediately to replace the old CRL.

To request immediate download of the latest CRL, use the following command in global configuration mode:

Command	Purpose
`Router(config)# crypto ca crl request` *name*	Requests an updated CRL. This command replaces the currently stored CRL at your router with the newest version of the CRL.

Querying a Certificate Revocation List

You can query a certificate revocation list (CRL) only when you configure your router with a trusted root. The following description and task apply only when your router has been configured with a trusted root.

When your router receives a certificate from a peer from another domain (with a different CA), the CRL downloaded from the CA of the router will not include certificate information about the peer. Therefore, you should check the CRL published by the configured root with the LDAP URL to ensure that the certificate of the peer has not been revoked.

If you would like CRL of the root certificate to be queried when the router reboots, you must enter the **crl query** command at this point.

To query the CRL published by the configured root with the LDAP URL, enter the following command in trusted root configuration mode:

Command	Purpose
`Router(ca-root)# crl query`	Queries the CRL published by the configured root with the LDAP URL. The URL used to query the CRL must be an LDAP URL. **Note** After you enter this command, an entry is created in the router for the **root subject name** command. The entry is based on information contained in the router.

Deleting RSA Keys from Your Router

Under certain circumstances you may want to delete your router's RSA keys. For example, if you believe the RSA keys were compromised in some way and should no longer be used, you should delete the keys.

To delete all RSA keys from your router, use the following command in global configuration mode:

Command	Purpose
`Router(config)# crypto key zeroize rsa`	Deletes all of your router's RSA keys.

After you delete a router's RSA keys, you should also complete these two additional tasks:

- Ask the CA administrator to revoke your router's certificates at the CA; you must supply the challenge password you created when you originally obtained the router's certificates with the **crypto ca enroll** command.
- Manually remove the router's certificates from the router configuration, as described in the section "Deleting Certificates from the Configuration."

Deleting a Peer's Public Keys

Under certain circumstances you may want to delete other peers' RSA public keys from your router's configuration. For example, if you no longer trust the integrity of a peer's public key, you should delete the key.

To delete a peer's RSA public key, use the following commands, beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# crypto key pubkey-chain rsa`	Enters public key configuration mode.

Step 2	`Router(config-pubkey-c)# no named-key` *key-name* `[encryption | signature]`
	
	or
	
	`Router(config-pubkey-c)# no addressed-key` *key-address* `[encryption | signature]`
		Deletes a remote peer's RSA public key. Specify the peer's fully qualified domain name or the remote peer's IP address.

Step 3	`exit`	Returns to global configuration mode.

Deleting Certificates from the Configuration

If the need arises, you can delete certificates that are saved at your router. Your router saves its own certificates, the certificate of the CA, and any RA certificates (unless you put the router into query mode as explained in the section "Managing NVRAM Memory Usage").

To delete your router's certificate or RA certificates from your router's configuration, use the following commands in global configuration mode:

	Command	Purpose
Step 1	`Router# show crypto ca certificates`	Displays the certificates stored on your router; note (or copy) the serial number of the certificate you wish to delete.
Step 2	`Router(config)# crypto ca certificate chain` *name*	Enters certificate chain configuration mode.
Step 3	`Router(config-cert-cha)# no certificate` *certificate-serial-number*	Deletes the certificate.

To delete the CA's certificate, you must remove the entire CA identity, which also removes all certificates associated with the CA—your router's certificate, the CA certificate, and any RA certificates.

To remove a CA identity, use the following command in global configuration mode:

Command	Purpose
`Router(config)# no crypto ca identity` *name*	Deletes all identity information and certificates associated with the CA.

Viewing Keys and Certificates

To view keys and certificates, use the following commands in EXEC mode:

	Command	Purpose
Step 1	`Router# show crypto key mypubkey rsa`	Displays your router's RSA public keys.
Step 2	`Router# show crypto key pubkey-chain rsa`	Displays a list of all the RSA public keys stored on your router. These include the public keys of peers who have sent your router their certificates during peer authentication for IPSec.
Step 3	`Router# show crypto key pubkey-chain rsa [name` *key-name* `|` `address` *key-address*`]`	Displays details of a particular RSA public key stored on your router.
Step 4	`Router# show crypto ca certificates`	Displays information about your certificate, the CA's certificate, and any RA certificates.
Step 5	`Router# show crypto ca roots`	Displays the CA roots configured in the router. **Note** This command can be implemented only when multiple CAs are configured in the router.

What to Do Next

After you have finished configuring this feature, you should configure IKE and IPSec. IKE configuration is described in the chapter "Configuring Internet Key Exchange Security Protocol." IPSec configuration is described in the chapter "Configuring IPSec Network Security."

CA Interoperability Configuration Examples

The following configuration is for a router named "myrouter." In this example, IPSec is configured and the IKE protocol and CA interoperability are configured in support of IPSec.

In this example, general-purpose RSA keys were generated, but you will notice that the keys are not saved or displayed in the configuration.

Comments are included within the configuration to explain various commands.

```
!
no service password-encryption
service udp-small-servers
service tcp-small-servers
!
! CA interoperability requires you to configure your router's hostname:
hostname myrouter
!
enable secret 5 <removed>
enable password <removed>
!
! CA interoperability requires you to configure your router's IP domain name:
ip domain-name example.com
ip name-server 172.29.2.132
ip name-server 192.168.30.32
!
! The following configures a transform set (part of IPSec configuration):
crypto ipsec transform-set my-transformset esp-3des esp-sha-hmac
!
! The following declares the CA. (In this example, the CA does not support an RA.)
crypto ca identity example.com
 enrollment url http://ca_server
!
! The following shows the certificates and CRLs stored at the router, including
! the CA certificate (shown first), the router's certificate (shown next)
! and a CRL (shown last).
crypto ca certificate chain example.com
! The following is the CA certificate
! received via the 'crypto ca authenticate' command:
 certificate ca 3051DF7169BEE31B821DFE4B3A338E5F
  30820182 3082012C A0030201 02021030 51DF7169 BEE31B82 1DFE4B3A 338E5F30
  0D06092A 864886F7 0D010104 05003042 31163014 06035504 0A130D43 6973636F
  20537973 74656D73 3110300E 06035504 0B130744 65767465 73743116 30140603
  55040313 0D434953 434F4341 2D554C54 5241301E 170D3937 31323032 30313036
  32385A17 0D393831 32303230 31303632 385A3042 31163014 06035504 0A130D43
  6973636F 20537973 74656D73 3110300E 06035504 0B130744 65767465 73743116
  30140603 55040313 0D434953 434F4341 2D554C54 5241305C 300D0609 2A864886
  F70D0101 01050003 4B003048 024100C1 B69D7BF6 34E4EE28 A84E0DC6 FCA4DEA8
  04D89E50 C5EBE862 39D51890 D0D4B732 678BDBF2 80801430 E5E56E7C C126E2DD
  DBE9695A DF8E5BA7 E67BAE87 29375302 03010001 300D0609 2A864886 F70D0101
  04050003 410035AA 82B5A406 32489413 A7FF9A9A E349E5B4 74615E05 058BA3CE
  7C5F00B4 019552A5 E892D2A3 86763A1F 2852297F C68EECE1 F41E9A7B 2F38D02A
  B1D2F817 3F7B
  quit
```

```
! The following is the router's certificate
! received via the 'crypto ca enroll' command:
certificate 7D28D4659D22C49134B3D1A0C2C9C8FC
  308201A6 30820150 A0030201 0202107D 28D4659D 22C49134 B3D1A0C2 C9C8FC30
  0D06092A 864886F7 0D010104 05003042 31163014 06035504 0A130D43 6973636F
  20537973 74656D73 3110300E 06035504 0B130744 65767465 73743116 30140603
  55040313 0D434953 434F4341 2D554C54 5241301E 170D3938 30343234 30303030
  30305A17 0D393930 34323432 33353935 395A302F 311D301B 06092A86 4886F70D
  01090216 0E73636F 742E6369 73636F2E 636F6D31 0E300C06 03550405 13053137
  41464230 5C300D06 092A8648 86F70D01 01010500 034B0030 48024100 A207ED75
  DE8A9BC4 980958B7 28ADF562 1371D043 1FC93C24 8E9F8384 4D1A2407 60CBD7EC
  B15BD782 A687CA49 883369BE B35A4219 8FE742B0 91CF76EE 07EC9E69 02030100
  01A33530 33300B06 03551D0F 04040302 05A03019 0603551D 11041230 10820E73
  636F742E 63697363 6F2E636F 6D300906 03551D13 04023000 300D0609 2A864886
  F70D0101 04050003 410085F8 A5AFA907 B38731A5 0195D921 D8C45EFD B6082C28
  04A88CEC E9EC6927 F24874E4 30C4D7E2 2686E0B5 77F197E4 F82A8BA2 1E03944D
  286B661F 0305DF5F 3CE7
  quit
! The following is a CRL received by the router (via the router's own action):
crl
  3081C530 71300D06 092A8648 86F70D01 01020500 30423116 30140603 55040A13
  0D436973 636F2053 79737465 6D733110 300E0603 55040B13 07446576 74657374
  31163014 06035504 03130D43 4953434F 43412D55 4C545241 170D3938 30333233
  32333232 31305A17 0D393930 34323230 30303030 305A300D 06092A86 4886F70D
  01010205 00034100 7AA83057 AC5E5C65 B9812549 37F11B7B 5CA4CAED 830B3955
  A4DDD268 F567E29A E4B34691 C2162BD1 0540D7E6 5D6650D1 81DBBF1D 788F1DAC
  BBF761B2 81FCC0F1
  quit
!
! The following is an IPSec crypto map (part of IPSec configuration):
crypto map map-to-remotesite 10 ipsec-isakmp
 set peer 172.21.114.196
 set transform-set my-transformset
 match address 124
!
!
interface Loopback0
 ip address 10.0.0.1 255.0.0.0
!
interface Tunnel0
 ip address 10.0.0.2 255.0.0.0
 ip mtu 1490
 no ip route-cache
 no ip mroute-cache
 tunnel source 10.10.0.1
 tunnel destination 172.21.115.119
!
interface FastEthernet0/0
 ip address 172.21.115.118 255.255.255.240
 no ip mroute-cache
 loopback
 no keepalive
 shutdown
 media-type MII
 full-duplex
!
! The IPSec crypto map is applied to interface Ethernet1/0:
interface Ethernet1/0
 ip address 172.21.114.197 255.255.255.0
 bandwidth 128
 no keepalive
 no fair-queue
 no cdp enable
 crypto map map-to-remotesite
```

```
!
crypto isakmp policy 15
 encryption 3des
 hash md5
 authentication rsa-sig
 group 2
 lifetime 5000
crypto isakmp policy 20
 authentication pre-share
 lifetime 10000
crypto isakmp key 1234567890 address 171.69.224.33
```

Multiple CAs Configuration Examples

The following examples show configuring multiple CAs (trusted roots) using SCEP or TFTP.

In this example, the configured trusted root is named “griffin”. The “griffin” trusted root is installed on the megatron server. The SCEP protocol and the root proxy URL are used to obtain the root certificate.

```
crypto ca trusted-root griffin
    root SCEP http://griffin:80
    root proxy http://megatron:8080
!
crypto ca authenticate griffin
Root certificate MD5 finger print:
8B4EC8C1 9308376F A0253C2A 34112AA6
% Do you accept this certificate? [yes/no]:y
```

In this example, the configured trusted root is named “banana”. Using TFTP, “banana” is installed on the “strawberry” server, and the filename is ca-cert/banana.

```
crypto ca trusted-root banana
    root tftp strawberry ca-cert/banana
!
crypto ca authenticate banana
Loading ca-cert/banana from 10.4.9.10 (via Ethernet0):!
[OK - 785/4096 bytes]
!
! Root certificate MD5 finger print:
F3F53FFB 925D052F 0C801EE7 89774ED3
% Do you accept this certificate? [yes/no]:y
Root certificate accepted.
```

coincidental.