Configuring Digital Certificates

This chapter describes how to configure digital certificates and includes the following sections:

Information About Digital Certificates

CAs are responsible for managing certificate requests and issuing digital certificates. A digital certificate includes information that identifies a user or device, such as a name, serial number, company, department, or IP address. A digital certificate also includes a copy of the public key for the user or device. A CA can be a trusted third party, such as VeriSign, or a private (in-house) CA that you establish within your organization.

Tip For an example of a scenario that includes certificate configuration and load balancing, see the following URL: https://supportforums.cisco.com/docs/DOC-5964.

This section includes the following topics:

Public Key Cryptography

Digital signatures, enabled by public key cryptography, provide a way to authenticate devices and 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 private key. The signature is attached to the data and sent to the receiver. The receiver applies the public key of the sender to the data. If the signature sent with the data matches the result of applying the public key to the data, the validity of the message is established.

This process relies on the receiver having a copy of the public key of the sender and a high degree of certainty that this key belongs to the sender, not to someone pretending to be the sender.

Obtaining the public key of a sender is normally handled externally or through an operation performed at installation. For example, most web browsers are configured with the root certificates of several CAs by default. For VPN, the IKE protocol, a component of IPsec, can use digital signatures to authenticate peer devices before setting up security associations.

Certificate Scalability

Without digital certificates, you must manually configure each IPsec peer for each peer with which it communicates; as a result, each new peer that you add to a network would require a configuration change on each peer with which it needs to communicate securely.

When you use digital certificates, each peer is enrolled with a CA. When two peers try to communicate, they exchange certificates and digitally sign data to authenticate each other. When a new peer is added to the network, you enroll that peer with a CA and none of the other peers need modification. When the new peer attempts an IPsec connection, certificates are automatically exchanged and the peer can be authenticated.

With a CA, a peer authenticates itself to the remote peer by sending a certificate to the remote peer and performing some public key cryptography. Each peer sends its unique certificate, which was issued by the CA. This process works because each certificate encapsulates the public key for the associated peer, each certificate is authenticated by the CA, and all participating peers recognize the CA as an authenticating authority. The process is called IKE with an RSA signature.

The peer can continue sending its certificate for multiple IPsec sessions, and to multiple IPsec peers, until the certificate expires. When its certificate expires, the peer administrator must obtain a new one from the CA.

CAs can also revoke certificates for peers that no longer participate in IPsec. Revoked certificates are not recognized as valid by other peers. Revoked certificates are listed in a CRL, which each peer may check before accepting a certificate from another peer.

Some CAs have an RA as part of their implementation. An RA is a server that acts as a proxy for the CA, so that CA functions can continue when the CA is unavailable.

Key Pairs

Key pairs are RSA keys, which have the following characteristics:

- RSA keys can be used for SSH or SSL.
- SCEP enrollment supports the certification of RSA keys.

- For the purposes of generating keys, the maximum key modulus for RSA keys is 2048 bits. The default size is 1024. Many SSL connections using identity certificates with RSA key pairs that exceed 1024 bits can cause a high CPU usage on the ASA and rejected clientless logins.
- For signature operations, the supported maximum key size is 4096 bits.
- You can generate a general purpose RSA key pair, used for both signing and encryption, or you can generate separate RSA key pairs for each purpose. Separate signing and encryption keys help to reduce exposure of the keys, because SSL uses a key for encryption but not signing. However, IKE uses a key for signing but not encryption. By using separate keys for each, exposure of the keys is minimized.

Trustpoints

Trustpoints let you manage and track CAs and certificates. A trustpoint is a representation of a CA or identity pair. A trustpoint includes the identity of the CA, CA-specific configuration parameters, and an association with one, enrolled identity certificate.

After you have defined a trustpoint, you can reference it by name in commands requiring that you specify a CA. You can configure many trustpoints.

Note If an ASA has multiple trustpoints that share the same CA, only one of these trustpoints sharing the CA can be used to validate user certificates. To control which trustpoint sharing a CA is used for validation of user certificates issued by that CA, use the **support-user-cert-validation** command.

For automatic enrollment, a trustpoint must be configured with an enrollment URL, and the CA that the trustpoint represents must be available on the network and must support SCEP.

You can export and import the keypair and issued certificates associated with a trustpoint in PKCS12 format. This format is useful to manually duplicate a trustpoint configuration on a different ASA.

Certificate Enrollment

The ASA needs a CA certificate for each trustpoint and one or two certificates for itself, depending upon the configuration of the keys used by the trustpoint. If the trustpoint uses separate RSA keys for signing and encryption, the ASA needs two certificates, one for each purpose. In other key configurations, only one certificate is needed.

The ASA supports automatic enrollment with SCEP and with manual enrollment, which lets you paste a base-64-encoded certificate directly into the terminal. For site-to-site VPNs, you must enroll each ASA. For remote access VPNs, you must enroll each ASA and each remote access VPN client.

Proxy for SCEP Requests

The ASA can proxy SCEP requests between AnyConnect and a third-party CA. The CA only needs to be accessible to the ASA if it is acting as the proxy. For the ASA to provide this service, the user must authenticate using any of the methods supported by AAA before the ASA sends an enrollment request. You can also use host scan and dynamic access policies to enforce rules of eligibility to enroll.

The ASA supports this feature only with an AnyConnect SSL or IKEv2 VPN session. It supports all SCEP-compliant CAs, including IOS CS, Windows Server 2003 CA, and Windows Server 2008 CA.

Clientless (browser-based) access does not support SCEP proxy, although WebLaunch—clientless-initiated AnyConnect—does support it.

The ASA does not support polling for certificates.

The ASA supports load balancing for this feature.

Revocation Checking

When a certificate is issued, it is valid for a fixed period of time. Sometimes a CA revokes a certificate before this time period expires; for example, because of security concerns or a change of name or association. CAs periodically issue a signed list of revoked certificates. Enabling revocation checking forces the ASA to check that the CA has not revoked a certificate each time that it uses the certificate for authentication.

When you enable revocation checking, the ASA checks certificate revocation status during the PKI certificate validation process, which can use either CRL checking, OCSP, or both. OCSP is only used when the first method returns an error (for example, indicating that the server is unavailable).

With CRL checking, the ASA retrieves, parses, and caches CRLs, which provide a complete list of revoked (and unrevoked) certificates with their certificate serial numbers. The ASA evaluates certificates according to CRLs, also called authority revocation lists, from the identity certificate up the chain of subordinate certificate authorities.

OCSP offers a more scalable method of checking revocation status in that it localizes certificate status through a validation authority, which it queries for status of a specific certificate.

Supported CA Servers

The ASA supports the following CA servers:

Cisco IOS CS, ASA Local CA, and third-party X.509 compliant CA vendors including, but not limited to:

- Baltimore Technologies
- Entrust
- Digicert
- Geotrust
- GoDaddy
- iPlanet/Netscape
- Microsoft Certificate Services
- RSA Keon
- Thawte
- VeriSign

CRLs

CRLs provide the ASA with one way of determining whether a certificate that is within its valid time range has been revoked by the issuing CA. CRL configuration is part of configuration of a trustpoint.

You can configure the ASA to make CRL checks mandatory when authenticating a certificate by using the **revocation-check crl** command. You can also make the CRL check optional by using the **revocation-check crl none** command, which allows the certificate authentication to succeed when the CA is unavailable to provide updated CRL data.

The ASA can retrieve CRLs from CAs using HTTP, SCEP, or LDAP. CRLs retrieved for each trustpoint are cached for a configurable amount of time for each trustpoint.

When the ASA has cached a CRL for longer than the amount of time it is configured to cache CRLs, the ASA considers the CRL too old to be reliable, or "stale." The ASA tries to retrieve a newer version of the CRL the next time that a certificate authentication requires a check of the stale CRL.

The ASA caches CRLs for an amount of time determined by the following two factors:

- The number of minutes specified with the **cache-time** command. The default value is 60 minutes.
- The NextUpdate field in the CRLs retrieved, which may be absent from CRLs. You control whether the ASA requires and uses the NextUpdate field with the **enforcenextupdate** command.

The ASA uses these two factors in the following ways:

- If the NextUpdate field is not required, the ASA marks CRLs as stale after the length of time defined by the **cache-time** command.
- If the NextUpdate field is required, the ASA marks CRLs as stale at the sooner of the two times specified by the **cache-time** command and the NextUpdate field. For example, if the **cache-time** command is set to 100 minutes and the NextUpdate field specifies that the next update is 70 minutes away, the ASA marks CRLs as stale in 70 minutes.

If the ASA has insufficient memory to store all CRLs cached for a given trustpoint, it deletes the least recently used CRL to make room for a newly retrieved CRL.

OCSP

OCSP provides the ASA with a way of determining whether a certificate that is within its valid time range has been revoked by the issuing CA. OCSP configuration is part of trustpoint configuration.

OCSP localizes certificate status on a validation authority (an OCSP server, also called the *responder*) which the ASA queries for the status of a specific certificate. This method provides better scalability and more up-to-date revocation status than does CRL checking, and helps organizations with large PKI installations deploy and expand secure networks.

Note The ASA allows a five-second time skew for OCSP responses.

You can configure the ASA to make OCSP checks mandatory when authenticating a certificate by using the **revocation-check ocsp** command. You can also make the OCSP check optional by using the **revocation-check ocsp none** command, which allows the certificate authentication to succeed when the validation authority is unavailable to provide updated OCSP data.

OCSP provides three ways to define the OCSP server URL. The ASA uses these servers in the following order:

1. The OCSP URL defined in a match certificate override rule by using the **match certificate** command).
2. The OCSP URL configured by using the **ocsp url** command.
3. The AIA field of the client certificate.

Note To configure a trustpoint to validate a self-signed OCSP responder certificate, you import the self-signed responder certificate into its own trustpoint as a trusted CA certificate. Then you configure the **match certificate** command in the client certificate validating trustpoint to use the trustpoint that includes the self-signed OCSP responder certificate to validate the responder certificate. Use the same procedure for

configuring validating responder certificates external to the validation path of the client certificate.

The OCSP server (responder) certificate usually signs the OCSP response. After receiving the response, the ASA tries to verify the responder certificate. The CA normally sets the lifetime of the OCSP responder certificate to a relatively short period to minimize the chance of being compromised. The CA usually also includes an ocsp-no-check extension in the responder certificate, which indicates that this certificate does not need revocation status checking. However, if this extension is not present, the ASA tries to check revocation status using the same method specified in the trustpoint. If the responder certificate is not verifiable, revocation checks fail. To avoid this possibility, use the **revocation-check none** command to configure the responder certificate validating trustpoint, and use the r**evocation-check ocsp** command to configure the client certificate.

The Local CA

The local CA performs the following tasks:

- Integrates basic certificate authority operation on the ASA.
- Deploys certificates.
- Provides secure revocation checking of issued certificates.
- Provides a certificate authority on the ASA for use with browser-based and client-based SSL VPN connections.
- Provides trusted digital certificates to users, without the need to rely on external certificate authorization.
- Provides a secure, in-house authority for certificate authentication and offers straightforward user enrollment by means of a website login.

Storage for Local CA Files

The ASA accesses and implements user information, issued certificates, and revocation lists using a local CA database. This database resides in local flash memory by default, or can be configured to reside on an external file system that is mounted and accessible to the ASA.

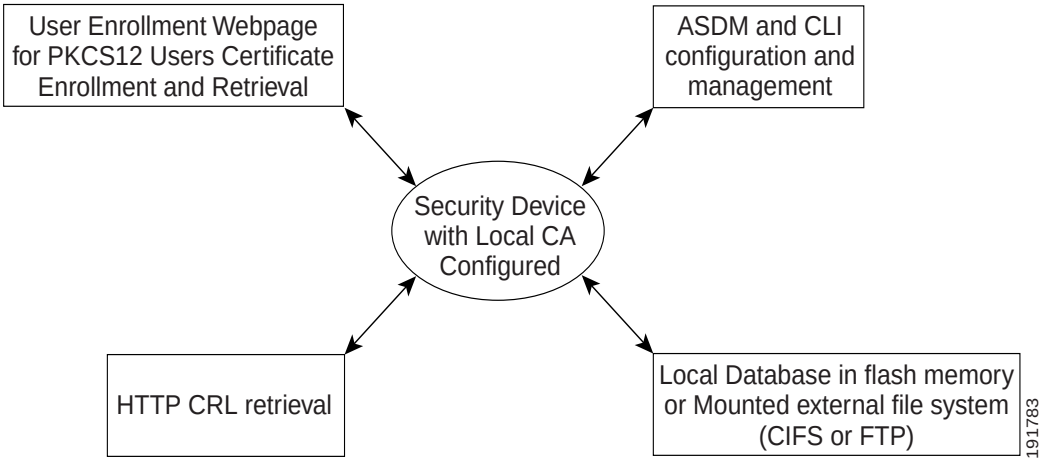
No limits exist on the number of users that can be stored in the local CA user database; however, if flash memory storage issues arise, syslogs are generated to alert the administrator to take action, and the local CA could be disabled until the storage issues are resolved. Flash memory can store a database with 3500 users or less; however, a database of more than 3500 users requires external storage.

The Local CA Server

After you configure a local CA server on the ASA, users can enroll for a certificate by logging into a website and entering a username and a one-time password that is provided by the local CA administrator to validate their eligibility for enrollment.

As shown in Figure 1-1, the local CA server resides on the ASA and handles enrollment requests from website users and CRL inquiries coming from other certificate validating devices and ASAs. Local CA database and configuration files are maintained either on the ASA flash memory (default storage) or on a separate storage device.

Figure 1-1 The Local CA



Licensing Requirements for Digital Certificates

The following table shows the licensing requirements for this feature:

Model	License Requirement
All models	Base License.

Prerequisites for Local Certificates

Local certificates have the following prerequisites:

- Make sure that the ASA is configured correctly to support certificates. An incorrectly configured ASA can cause enrollment to fail or request a certificate that includes inaccurate information.
- Make sure that the hostname and domain name of the ASA are configured correctly. To view the currently configured hostname and domain name, enter the **show running-config** command. For information about configuring the hostname and domain name, see the "Configuring the Hostname, Domain Name, and Passwords" section on page 1-1.
- Make sure that the ASA clock is set accurately before configuring the CA. Certificates have a date and time that they become valid and expire. When the ASA enrolls with a CA and obtains a certificate, the ASA checks that the current time is within the valid range for the certificate. If it is outside that range, enrollment fails. For information about setting the clock, see the "Setting the Date and Time" section on page 1-4.

Prerequisites for SCEP Proxy Support

Configuring the ASA as a proxy to submit requests for third-party certificates has the following requirements:

- AnyConnect Secure Mobility Client 3.0 or later must be running at the endpoint.

- The authentication method, configured in the connection profile for your group policy, must be set to use both AAA and certificate authentication.
- An SSL port must be open for IKEv2 VPN connections.
- The CA must be in auto-grant mode.

Guidelines and Limitations

This section includes the guidelines and limitations for this feature.

Context Mode Guidelines

- Supported in single and multiple context mode for a local CA.
- Supported in single context mode only for third-party CAs.

Firewall Mode Guidelines

Supported in routed and transparent firewall mode.

Failover Guidelines

- Does not support replicating sessions in Stateful Failover.
- Does not support failover for local CAs.

IPv6 Guidelines

Does not support IPv6.

Additional Guidelines

- For ASAs that are configured as CA servers or clients, limit the validity period of the certificate to less than the recommended end date of 03:14:08 UTC, January 19, 2038. This guideline also applies to imported certificates from third-party vendors.
- You cannot configure the local CA when failover is enabled. You can only configure the local CA server for standalone ASAs without failover. For more information, see CSCty43366.
- When a certificate enrollment is completed, the ASA stores a PKCS12 file containing the user's keypair and certificate chain, which requires about 2 KB of flash memory or disk space per enrollment. The actual amount of disk space depends on the configured RSA key size and certificate fields. Keep this guideline in mind when adding a large number of pending certificate enrollments on an ASA with a limited amount of available flash memory, because these PKCS12 files are stored in flash memory for the duration of the configured enrollment retrieval timeout.
- The **lifetime ca-certificate** command takes effect when the local CA server certificate is first generated (that is, when you initially configure the local CA server and issue the **no shutdown** command). When the CA certificate expires, the configured lifetime value is used to generate the new CA certificate. You cannot change the lifetime value for existing CA certificates.
- You should configure the ASA to use an identity certificate to protect ASDM traffic and HTTPS traffic to the management interface. Identity certificates that are automatically generated with SCEP are regenerated after each reboot, so make sure that you manually install your own identity certificates. For an example of this procedure that applies only to SSL, see the following URL: http://www.cisco.com/en/US/products/ps6120/products_configuration_example09186a00809fcf91.shtml.

- The ASA and the AnyConnect clients can only validate certificates in which the X520Serialnumber field (the serial number in the Subject Name) is in PrintableString format. If the serial number format uses encoding such as UTF8, the certificate authorization will fail.
- Use only valid characters and values for the certificate parameters when you import them on the ASA.
- To use a wildcard (*) symbol, make sure that you use encoding on the CA server that allows this character in the string value. Although RFC 5280 recommends using either a UTF8String or PrintableString, you should use UTF8String because PrintableString does not recognize the wildcard as a valid character. The ASA rejects the imported certificate if an invalid character or value is found during the import. For example:

```
ERROR: Failed to parse or verify imported certificate ciscoasa(config)# Read
162*H÷ytes as CA certificate:0U0= \Ivr"phÕV°3é¼þ0 CRYPTO_PKI(make trustedCerts list)
CERT-C: E ../cert-c/source/certlist.c(302) : Error #711h
CRYPTO_PKI: Failed to verify the ID certificate using the CA certificate in trustpoint
mm.
CERT-C: E ../cert-c/source/p7contnt.c(169) : Error #703h
crypto_certc_pkcs7_extract_certs_and_crls failed (1795):
crypto_certc_pkcs7_extract_certs_and_crls failed
CRYPTO_PKI: status = 1795: failed to verify or insert the cert into storage
```

Configuring Digital Certificates

This section describes how to configure local CA certificates. Make sure that you follow the sequence of tasks listed to correctly configure this type of digital certificate. This section includes the following topics:

Configuring Key Pairs

To generate key pairs, perform the following steps:

	Command	Purpose
Step 1	**crypto key generate rsa** **Example:** `hostname/contexta(config)# crypto key generate rsa`	Generates one, general-purpose RSA key pair. The default key modulus is 1024. To specify other modulus sizes, use the **modulus** keyword. **Note** Many SSL connections using identity certificates with RSA key pairs that exceed 1024 bits can cause high CPU usage on the ASA and rejected clientless logins.
Step 2	**crypto key generate rsa label** *key-pair-label* **Example:** `hostname/contexta(config)# crypto key generate rsa label exchange`	(Optional) Assigns a label to each key pair. The label is referenced by the trustpoint that uses the key pair. If you do not assign a label, the key pair is automatically labeled, *Default-RSA-Key*.
Step 3	**show crypto key** *name of key* **Example:** `hostname/contexta(config)# show crypto key examplekey`	Verifies key pairs that you have generated.
Step 4	**write memory** **Example:** `hostname(config)# write memory`	Saves the key pair that you have generated.

Removing Key Pairs

To remove key pairs, perform the following steps:

Command	Purpose
crypto key zeroize rsa **Example:** `hostname(config)# crypto key zeroize rsa`	Removes key pairs.

Examples

The following example shows how to remove key pairs:

```
hostname(config)# crypto key zeroize rsa
WARNING: All RSA keys will be removed.
WARNING: All device certs issued using these keys will also be removed.

Do you really want to remove these keys? [yes/no] y
```

Configuring Trustpoints

To configure a trustpoint, perform the following steps:

	Command	Purpose
Step 1	**crypto ca trustpoint** *trustpoint-name* **Example:** `hostname/contexta(config)# crypto ca trustpoint Main`	Creates a trustpoint that corresponds to the CA from which the ASA needs to receive a certificate. Enters the crypto ca trustpoint configuration mode, which controls CA-specific trustpoint parameters that you may configure starting in Step 3. **Note** When you try to connect, a warning occurs to indicate that the trustpoint does not contain an ID certificate when an attempt is made to retrieve the ID certificate from the trustpoint.
Step 2	Choose one of the following options:	
	enrollment url *url* **Example:** `hostname/contexta(config-ca-trustpoint)# enrollment url http://10.29.67.142:80/certsrv/mscep/mscep.dll`	Requests automatic enrollment using SCEP with the specified trustpoint and configures the enrollment URL.
	enrollment terminal **Example:** `hostname/contexta(config-ca-trustpoint)# enrollment terminal`	Requests manual enrollment with the specified trustpoint by pasting the certificate received from the CA into the terminal.

Step 3	**revocation-check crl none**
	`revocation-check crl`
	`revocation-check none`
	
	Example:
	`hostname/contexta(config-ca-trustpoint)# revocation-check crl none`
	`hostname/contexta(config-ca-trustpoint)# revocation-check crl`
	`hostname/contexta(config-ca-trustpoint)# revocation-check none`
		Specifies the available CRL configuration options.
		
		Note To enable either required or optional CRL checking, make sure that you configure the trustpoint for CRL management after obtaining certificates.

Step 4	**crl configure** **Example:** `hostname/contexta(config-ca-trustpoint)# crl configure`	Enters crl configuration mode.
Step 5	**email** *address* **Example:** `hostname/contexta(config-ca-trustpoint)# email example.com`	During enrollment, asks the CA to include the specified e-mail address in the Subject Alternative Name extension of the certificate.

	Command	Purpose
Step 6	**enrollment retry period** **Example:** `hostname/contexta(config-ca-trustpoint)# enrollment retry period 5`	(Optional) Specifies a retry period in minutes, and applies only to SCEP enrollment.
Step 7	**enrollment retry count** **Example:** `hostname/contexta(config-ca-trustpoint)# enrollment retry period 2`	(Optional) Specifies a maximum number of permitted retries, and applies only to SCEP enrollment.
Step 8	**fqdn** *fqdn* **Example:** `hostname/contexta(config-ca-trustpoint)# fqdn example.com`	During enrollment, asks the CA to include the specified fully qualified domain name in the Subject Alternative Name extension of the certificate.
Step 9	**ip-address** *ip-address* **Example:** `hostname/contexta(config-ca-trustpoint)# ip-address 10.10.100.1`	During enrollment, asks the CA to include the IP address of the ASA in the certificate.
Step 10	**keypair** *name* **Example:** `hostname/contexta(config-ca-trustpoint)# keypair exchange`	Specifies the key pair whose public key is to be certified.
Step 11	**match certificate map-name override ocsp** **Example:** `hostname/contexta(config-ca-trustpoint)# match certificate examplemap override ocsp`	Configures OCSP URL overrides and trustpoints to use for validating OCSP responder certificates.
Step 12	**ocsp disable-nonce** **Example:** `hostname/contexta(config-ca-trustpoint)# ocsp disable-nonce`	Disables the nonce extension on an OCSP request. The nonce extension cryptographically binds requests with responses to avoid replay attacks.
Step 13	**ocsp url** **Example:** `hostname/contexta(config-ca-trustpoint)# ocsp url`	Configures an OCSP server for the ASA to use to check all certificates associated with a trustpoint rather than the server specified in the AIA extension of the client certificate.
Step 14	**password** *string* **Example:** `hostname/contexta(config-ca-trustpoint)# password mypassword`	Specifies a challenge phrase that is registered with the CA during enrollment. The CA usually uses this phrase to authenticate a subsequent revocation request.

	Command	Purpose
Step 15	**revocation check** **Example:** `hostname/contexta(config-ca-trustpoint)# revocation check`	Sets one or more methods for revocation checking: CRL, OCSP, and none.
Step 16	**subject-name** *X.500 name* **Example:** `hostname/contexta(config-ca-trustpoint)# myname X.500 examplename`	During enrollment, asks the CA to include the specified subject DN in the certificate. If a DN string includes a comma, enclose the value string within double quotes (for example, O=”Company, Inc.”).
Step 17	**serial-number** **Example:** `hostname/contexta(config-ca-trustpoint)# serial number JMX1213L2A7`	During enrollment, asks the CA to include the ASA serial number in the certificate.
Step 18	**write memory** **Example:** `hostname/contexta(config)# write memory`	Saves the running configuration.

Configuring CRLs for a Trustpoint

To use mandatory or optional CRL checking during certificate authentication, you must configure CRLs for each trustpoint. To configure CRLs for a trustpoint, perform the following steps:

	Command	Purpose
Step 1	**crypto ca trustpoint** *trustpoint-name* **Example:** `hostname (config)# crypto ca trustpoint Main`	Enters crypto ca trustpoint configuration mode for the trustpoint whose CRL configuration you want to modify. **Note** Make sure that you have enabled CRLs before entering this command. In addition, the CRL must be available for authentication to succeed.
Step 2	**crl configure** **Example:** `hostname (config-ca-trustpoint)# crl configure`	Enters crl configuration mode for the current trustpoint. **Tip** To set all CRL configuration parameters to default values, use the **default** command. At any time during CRL configuration, reenter this command to restart the procedure.
Step 3	Do one of the following:	

	Command	Purpose
	policy cdp **Example:** `hostname (config-ca-crl)# policy cdp`	Configures retrieval policy. CRLs are retrieved only from the CRL distribution points specified in authenticated certificates. **Note** SCEP retrieval is not supported by distribution points specified in certificates. To continue, go to Step 5.
	policy static **Example:** `hostname (config-ca-crl)# policy static`	Configures retrieval policy. CRLs are retrieved only from URLs that you configure. To continue, go to Step 4.
	policy both **Example:** `hostname (config-ca-crl)# policy both`	Configures retrieval policy. CRLs are retrieved from CRL distribution points specified in authenticated certificates and from URLs that you configure. To continue, go to Step 4.
Step 4	**url** *n url* **Example:** `hostname (config-ca-crl)# url 2 http://www.example.com`	If you used the keywords **static** or **both** when you configured the CRL policy, you must configure URLs for CRL retrieval. You can enter up to five URLs, ranked 1 through 5. The *n* is the rank assigned to the URL. To remove a URL, use the **no url** *n* command.
Step 5	**protocol http** | **ldap** | **scep** **Example:** `hostname (config-ca-crl)# protocol http`	Configures the retrieval method. Specifies HTTP, LDAP, or SCEP as the CRL retrieval method.
Step 6	**cache-time** *refresh-time* **Example:** `hostname (config-ca-crl)# cache-time 420`	Configures how long the ASA caches CRLs for the current trustpoint. *refresh-time* is the number of minutes that the ASA waits before considering a CRL stale.
Step 7	Do one of the following:	
	enforcenextupdate **Example:** `hostname (config-ca-crl)# enforcenextupdate`	Requires the NextUpdate field in CRLs. This is the default setting.
	no enforcenextupdate **Example:** `hostname (config-ca-crl)# no enforcenextupdate`	Allows the NextUpdate field to be absent in CRLs.

	Command	Purpose
Step 8	**ldap-defaults** *server* **Example:** `hostname (config-ca-crl)# ldap-defaults ldap1`	Identifies the LDAP server to the ASA if LDAP is specified as the retrieval protocol. You can specify the server by DNS hostname or by IP address. You can also provide a port number if the server listens for LDAP queries on a port other than the default of 389. **Note** If you use a hostname instead of an IP address to specify the LDAP server, make sure that you have configured the ASA to use DNS.
Step 9	**ldap-dn** *admin-DN password* **Example:** `hostname (config-ca-crl)# ldap-dn cn=admin,ou=devtest,o=engineering c00lRunZ`	Allows CRL retrieval if the LDAP server requires credentials.
Step 10	**crypto ca crl request** *trustpoint* **Example:** `hostname (config-ca-crl)# crypto ca crl request Main`	Retrieves the current CRL from the CA represented by the specified trustpoint and tests the CRL configuration for the current trustpoint.
Step 11	**write memory** **Example:** `hostname (config)# write memory`	Saves the running configuration.

Exporting a Trustpoint Configuration

To export a trustpoint configuration, enter the following command:

Command	Purpose
crypto ca export *trustpoint* **Example:** `hostname(config)# crypto ca export Main`	Exports a trustpoint configuration with all associated keys and certificates in PKCS12 format. The ASA displays the PKCS12 data in the terminal. You can copy the data. The trustpoint data is password protected; however, if you save the trustpoint data in a file, make sure that the file is in a secure location.

Examples

The following example exports PKCS12 data for the trustpoint Main with the passphrase Wh0zits:

```
hostname (config)# crypto ca export Main pkcs12 Wh0zits

Exported pkcs12 follows:

[ PKCS12 data omitted ]

---End - This line not part of the pkcs12---
```

Importing a Trustpoint Configuration

To import a trustpoint configuration, enter the following command:

Command	Purpose
crypto ca import *trustpoint* **pkcs12** **Example:** `hostname(config)# crypto ca import Main pkcs12`	Imports keypairs and issued certificates that are associated with a trustpoint configuration. The ASA prompts you to paste the text into the terminal in base 64 format. The key pair imported with the trustpoint is assigned a label that matches the name of the trustpoint that you create. **Note** If an ASA has trustpoints that share the same CA, you can use only one of the trustpoints that share the CA to validate user certificates. To control which trustpoint that shares a CA is used for validation of user certificates issued by that CA, use the **support-user-cert-validation** keyword.

Examples

The following example manually imports PKCS12 data to the trustpoint Main with the passphrase Wh0zits:

```
hostname (config)# crypto ca import Main pkcs12 Wh0zits

Enter the base 64 encoded pkcs12.
End with a blank line or the word "quit" on a line by itself:
[ PKCS12 data omitted ]
quit
INFO: Import PKCS12 operation completed successfully
```

The following example manually imports a certificate for the trustpoint Main:

```
hostname (config)# crypto ca import Main certificate
% The fully-qualified domain name in the certificate will be:
securityappliance.example.com

Enter the base 64 encoded certificate.
End with a blank line or the word "quit" on a line by itself
[ certificate data omitted ]
quit
INFO: Certificate successfully imported
```

Configuring CA Certificate Map Rules

You can configure rules based on the Issuer and Subject fields of a certificate. Using the rules you create, you can map IPsec peer certificates to tunnel groups with the **tunnel-group-map** command. The ASA supports one CA certificate map, which can include many rules.

To configure a CA certificate map rule, perform the following steps:

	Command	Purpose
Step 1	**crypto ca certificate map** *sequence-number* **Example:** `hostname(config)# crypto ca certificate map 1`	Enters CA certificate map configuration mode for the rule you want to configure and specifies the rule index number.
Step 2	**issuer-name** *DN-string* **Example:** `hostname(config-ca-cert-map)# issuer-name cn=asa.example.com`	Specifies the distinguished name of all issued certificates. which is also the subject-name DN of the self-signed CA certificate. Use commas to separate attribute-value pairs. Insert quotation marks around any value that includes a comma. An issuer-name must be less than 500 alphanumeric characters. The default issuer-name is cn=*hostame.domain-name.*
Step 3	**subject-name attr** *tag* **eq** | **co** | **ne** | **nc** *string* **Example:** `hostname(config-ca-cert-map)# subject-name attr cn eq mycert`	Specifies tests that the ASA can apply to values found in the Subject field of certificates. The tests can apply to specific attributes or to the entire field. You can configure many tests per rule, and all the tests you specify with these commands must be true for a rule to match a certificate. The following are valid operators: • eq—The field or attribute must be identical to the value given. • ne—The field or attribute cannot be identical to the value given. • co—Part or all of the field or attribute must match the value given. • nc—No part of the field or attribute can match the value given.
Step 4	**write memory** **Example:** `hostname (config)# write memory`	Saves the running configuration.

Obtaining Certificates Manually

To obtain certificates manually, perform the following steps:

	Command	Purpose
Step 1	**crypto ca authenticate** *trustpoint* **Example:** hostname(config)# crypto ca authenticate Main Enter the base 64 encoded CA certificate. End with a blank line or the word "quit" on a line by itself MIIDRTCCAu+gAwIBAgIQKVcqP/KW74VP0NZzL+JbRTANBgkqhkiG 9w0BAQUFADCB *[certificate data omitted]* /7QEM8izy0EOTSErKu7Nd76jwf5e4qttkQ== **quit** INFO: Certificate has the following attributes: Fingerprint: 24b81433 409b3fd5 e5431699 8d490d34 Do you accept this certificate? [yes/no]: **y** Trustpoint CA certificate accepted. % Certificate successfully imported	Imports the CA certificate for the configured trustpoint. **Note** This step assumes that you have already obtained a base-64 encoded CA certificate from the CA represented by the trustpoint. Whether a trustpoint requires that you manually obtain certificates is determined by the use of the **enrollment terminal** command when you configure the trustpoint. For more information, see the "Configuring Trustpoints" section on page 1-11.
Step 2	**crypto ca enroll** *trustpoint* **Example:** hostname(config)# crypto ca enroll Main % Start certificate enrollment .. % The fully-qualified domain name in the certificate will be: securityappliance.example.com % Include the device serial number in the subject name? [yes/no]: **n** Display Certificate Request to terminal? [yes/no]: **y** Certificate Request follows: MIIBoDCCAQkCAQAwIzEhMB8GCSqGSIb3DQEJAhYSRmVyYWxQaXgu Y2lzY28uY29t *[certificate request data omitted]* jF4waw68eOxQxVmdgMWeQ+RbIOYmvt8g6hnBTrd0GdqjjVLt ---End - This line not part of the certificate request--- Redisplay enrollment request? [yes/no]: **n**	Enrolls the ASA with the trustpoint. Generates a certificate for signing data and depending on the type of keys that you have configured, for encrypting data. If you use separate RSA keys for signing and encryption, the **crypto ca enroll** command displays two certificate requests, one for each key. If you use general-purpose RSA keys for both signing and encryption, the **crypto ca enroll** command displays one certificate request. To complete enrollment, obtain a certificate for all certificate requests generated by the **crypto ca enroll** command from the CA represented by the applicable trustpoint. Make sure that the certificate is in base-64 format.

	Command	Purpose

Step 3	**crypto ca import** *trustpoint* **certificate**
	
	Example:
	`hostname (config)# crypto ca import Main certificate`
	`% The fully-qualified domain name in the certificate will be: securityappliance.example.com`
	
	`Enter the base 64 encoded certificate.`
	`End with a blank line or the word "quit" on a line by itself`
	`[ certificate data omitted ]`
	`quit`
	`INFO: Certificate successfully imported`
		Imports each certificate you receive from the CA. Requests that you paste the certificate to the terminal in base-64 format.

Step 4	**show crypto ca server certificate** **Example:** `hostname(config)# show crypto ca server certificate Main`	Verifies that the enrollment process was successful by displaying certificate details issued for the ASA and the CA certificate for the trustpoint.
Step 5	**write memory** **Example:** `hostname(config)# write memory`	Saves the running configuration. Repeat these steps for each trustpoint that you configure for manual enrollment.

Obtaining Certificates Automatically with SCEP

To obtain certificates automatically using SCEP, perform the following steps:

	Command	Purpose
Step 1	**crypto ca authenticate** *trustpoint* **Example:** `hostname/contexta(config)# crypto ca authenticate Main`	Obtains the CA certificate for the configured trustpoint. **Note** This step assumes that you have already obtained a base-64 encoded CA certificate from the CA represented by the trustpoint. When you configure the trustpoint, use of the **enrollment url** command determines whether or not you must obtain certificates automatically via SCEP. For more information, see the "Configuring Trustpoints" section on page 1-11.
Step 2	**crypto ca enroll** *trustpoint* **Example:** `hostname/contexta(config)# crypto ca enroll Main`	Enrolls the ASA with the trustpoint. Retrieves a certificate for signing data and depending on the type of keys that you have configured, for encrypting data. Before entering this command, contact the CA administrator, who may need to authenticate the enrollment request manually before the CA grants certificates. If the ASA does not receive a certificate from the CA within one minute (the default) of sending a certificate request, it resends the certificate request. The ASA continues sending a certificate request each minute until a certificate is received. If the fully qualified domain name configured for the trustpoint is not identical to the fully qualified domain name of the ASA, including the case of the characters, a warning appears. To resolve this issue, exit the enrollment process, make any necessary corrections, and reenter the **crypto ca enroll** command. **Note** If the ASA reboots after you have issued the **crypto ca enroll** command but before you have received the certificate, reenter the **crypto ca enroll** command and notify the CA administrator.

	Command	Purpose
Step 3	**show crypto ca server certificate** **Example:** `hostname/contexta(config)# show crypto ca server certificate Main`	Verifies that the enrollment process was successful by displaying certificate details issued for the ASA and the CA certificate for the trustpoint.
Step 4	**write memory** **Example:** `hostname/contexta(config)# write memory`	Saves the running configuration.

Configuring Proxy Support for SCEP Requests

To configure the ASA to authenticate remote access endpoints using third-party CAs, perform the following steps:

	Command	Purpose
Step 1	**crypto ikev2 enable outside client-services port** *portnumber* **Example:** `hostname(config-tunnel-ipsec)# crypto ikev2 enable outside client-services`	Enables client services. **Note** Needed only if you support IKEv2. Enter this command in tunnel-group ipsec-attributes configuration mode. The default port number is 443.

Step 2	**scep-enrollment enable**
	
	Example:
	`hostname(config-tunnel-general)# scep-enrollment enable`
	`INFO: 'authentication aaa certificate' must be configured to complete setup of this option.`
		Enables SCEP enrollment for the tunnel group.
		Enter this command in tunnel-group general-attributes configuration mode.

Step 3	**scep-forwarding-url value** *URL* **Example:** `hostname(config-group-policy)# scep-forwarding-url value http://ca.example.com:80/`	Enrolls the SCEP CA for the group policy. Enter this command once per group policy to support a third-party digital certificate. Enter the command in group-policy general-attributes configuration mode. *URL* is the SCEP URL on the CA.

Step 4	**secondary-pre-fill-username clientless hide use-common-password** *password*
	
	Example:
	`hostname(config)# tunnel-group remotegrp webvpn-attributes`
	`hostname(config-tunnel-webvpn)# secondary-pre-fill-username clientless hide use-common-password secret`
		Supplies a common, secondary password when a certificate is unavailable for WebLaunch support of the SCEP proxy.
		You must use the **hide** keyword to support the SCEP proxy.
		For example, a certificate is not available to an endpoint requesting one. Once the endpoint has the certificate, AnyConnect disconnects, then reconnects to the ASA to qualify for a DAP policy that provides access to internal network resources.

	Command	Purpose
Step 5	**secondary-pre-fill-username ssl-client hide use-common-password** *password* **Example:** `hostname(config-tunnel-webvpn)# secondary-pre-fill-username ssl-client hide use-common-password secret`	Hides the secondary prefill username for AnyConnect VPN sessions. Despite the **ssl-client** keyword inherited from earlier releases, use this command to support AnyConnect sessions that use either IKEv2 or SSL. You must use the **hide** keyword to support the SCEP proxy.
Step 6	**secondary-username-from-certificate** {**use-entire-name** | **use-script** | {*primary_attr* [*secondary-attr*]}} [**no-certificate-fallback cisco-secure-desktop machine-unique-id**] **Example:** `hostname(config-tunnel-webvpn)# secondary-username-from-certificate CN no-certificate-fallback cisco-secure-desktop machine-unique-id`	Supplies the username when a certificate is unavailable.

Enabling the Local CA Server

Before enabling the local CA server, you must first create a passphrase of at least seven characters to encode and archive a PKCS12 file that includes the local CA certificate and keypair to be generated. The passphrase unlocks the PKCS12 archive if the CA certificate or keypair is lost.

To enable the local CA server, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**no shutdown** **Example:** `hostname (config-ca-server)# no shutdown`	Enables the local CA server. Generates the local CA server certificate, keypair and necessary database files, and archives the local CA server certificate and keypair to storage in a PKCS12 file. Requires an 8-65 alphanumeric character password. After initial startup, you can disable the local CA without being prompted for the passphrase. **Note** After you enable the local CA server, save the configuration to make sure that the local CA certificate and keypair are not lost after a reboot occurs.

Examples

The following example enables the local CA server:

```
hostname (config)# crypto ca server
```

```
hostname (config-ca-server)# no shutdown

% Some server settings cannot be changed after CA certificate generation.
% Please enter a passphrase to protect the private key
% or type Return to exit

Password: caserver

Re-enter password: caserver

Keypair generation process begin. Please wait...
```

The following is sample output that shows local CA server configuration and status:

```
Certificate Server LOCAL-CA-SERVER:
    Status: enabled
    State: enabled
    Server's configuration is locked (enter "shutdown" to unlock it)
    Issuer name: CN=wz5520-1-16
    CA certificate fingerprint/thumbprint: (MD5)
        76dd1439 ac94fdbc 74a0a89f cb815acc
    CA certificate fingerprint/thumbprint: (SHA1)
        58754ffd 9f19f9fd b13b4b02 15b3e4be b70b5a83
    Last certificate issued serial number: 0x6
    CA certificate expiration timer: 14:25:11 UTC Jan 16 2008
    CRL NextUpdate timer: 16:09:55 UTC Jan 24 2007
    Current primary storage dir: flash:
```

Configuring the Local CA Server

To configure the local CA server, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Generates the local CA.
Step 2	**smtp from-address** *e-mail_address* **Example:** `hostname (config-ca-server) # smtp from-address SecurityAdmin@example.com`	Specifies the SMTP from-address, a valid e-mail address that the local CA uses as a from address when sending e-mail messages that deliver OTPs for an enrollment invitation to users.

	Command	Purpose
Step 3	**subject-name-default** *dn* **Example:** `hostname (config-ca-server)# subject-name-default cn=engineer, o=asc systems, c="US"`	(Optional) Specifies the subject-name DN that is appended to each username on issued certificates. The subject-name DN and the username combine to form the DN in all user certificates that are issued by the local CA server. If you do not specify a subject-name DN, you must specify the exact subject name DN to be included in a user certificate each time that you add a user to the user database. **Note** Make sure that you review all optional parameters carefully before you enable the configured local CA, because you cannot change issuer-name and keysize server values after you enable the local CA for the first time.
Step 4	**no shutdown** **Example:** `hostname (config-ca-server)# no shutdown`	Creates the self-signed certificate and associates it with the local CA on the ASA. The self-signed certificate key usage extension has key encryption, key signature, CRL signing, and certificate signing capabilities. **Note** After the self-signed local CA certificate has been generated, to change any characteristics, you must delete the existing local CA server and completely recreate it. The local CA server keeps track of user certificates, so the administrator can revoke or restore privileges as needed.

Examples

The following example shows how to configure and enable the local CA server using the predefined default values for all required parameters:

```
hostname (config)# crypto ca server
hostname (config-ca-server) # smtp from-address SecurityAdmin@example.com
hostname (config-ca-server)# subject-name-default cn=engineer, o=asc Systems, c=US
hostname (config-ca-server)# no shutdown
```

Customizing the Local CA Server

To configure a customized local CA server, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**issuer-name** *DN-string* **Example:** `hostname (config-ca-server)# issuer-name cn=xx5520,cn=30.132.0.25,ou=DevTest,ou=QA,o=ASC Systems`	Specifies parameters that do not have default values.
Step 3	**smtp subject** *subject-line* **Example:** `hostname (config-ca-server) # smtp subject Priority E-Mail: Enclosed Confidential Information is Required for Enrollment`	Customizes the text that appears in the subject field of all e-mail messages sent from the local CA server
Step 4	**smtp from-address** *e-mail_address* **Example:** `hostname (config-ca-server) # smtp from-address SecurityAdmin@example.com`	**S**pecifies the e-mail address that is to be used as the From: field of all e-mail messages that are generated by the local CA server.
Step 5	**subject-name-default** *dn* **Example:** `hostname (config-ca-server) # subject-name default cn=engineer, o=ASC Systems, c=US`	Specifies an optional subject-name DN to be appended to a username on issued certificates. The default subject-name DN becomes part of the username in all user certificates issued by the local CA server. The allowed DN attribute keywords are as follows: • C = Country • CN = Common Name • EA = E-mail Address • L = Locality • O = Organization Name • OU = Organization Unit • ST = State/Province • SN = Surname • ST = State/Province **Note** If you do not specify a subject-name-default to serve as a standard subject-name default, you must specify a DN each time that you add a user.

Debugging the Local CA Server

To debug the newly configured local CA server, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**debug crypto ca server** **Example:** `hostname (config-ca-server)# debug crypto ca server`	Displays debugging messages when you configure and enable the local CA server. Performs level 1 debugging functions; levels 1-255 are available. **Note** Debugging commands might slow down traffic on busy networks. Levels 5 and higher are reserved for raw data dumps and should be avoided during normal debugging because of excessive output.

Disabling the Local CA Server

To disable the local CA server, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.

Step 2	**shutdown**
	
	Example:
	`hostname (config-ca-server)# shutdown`
	`INFO: Local CA Server has been shutdown.`
		Disables the local CA server. Disables website enrollment and allows you to modify the local CA server configuration. Stores the current configuration and associated files. After initial startup, you can reenable the local CA without being prompted for the passphrase.

Deleting the Local CA Server

To delete an existing local CA server (either enabled or disabled), enter one of the following commands:

Command	Purpose
Do one of the following:	

Command	Purpose
no crypto ca server **Example:** `hostname (config)# no crypto ca server`	Removes an existing local CA server (either enabled or disabled). **Note** Deleting the local CA server removes the configuration from the ASA. After the configuration has been deleted, it is unrecoverable. Make sure that you also delete the associated local CA server database and configuration files (that is, all files with the wildcard name, LOCAL-CA-SERVER.*).
clear configure crypto ca server **Example:** `hostname (config)# clear config crypto ca server`	

Configuring Local CA Certificate Characteristics

You can configure the following characteristics of local CA certificates:

- The name of the certificate issuer as it appears on all user certificates.
- The lifetime of the local CA certificates (server and user) and the CRL.
- The length of the public and private keypairs associated with local CA and user certificates.

This section includes the following topics:

Configuring the Issuer Name

To configure the certificate issuer name, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**issuer-name** *DN-string* **Example:** `hostname (config-ca-server)# issuer-name CN=xx5520,CN=30.132.0.25,ou=DevTest,ou=QA,O=ABC Systems`	Specifies the local CA certificate subject name. The configured certificate issuer name is both the subject name and issuer name of the self-signed local CA certificate, as well as the issuer name in all issued client certificates and in the issued CRL. The default issuer name in the local CA is in the format, *hostname.domainname*. **Note** You cannot change the issuer name value after the local CA is first enabled.

Configuring the CA Certificate Lifetime

To configure the local CA server certificate lifetime, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.

	Command	Purpose
Step 2	**lifetime ca-certificate** *time* **Example:** `hostname (config-ca-server)# lifetime ca-certificate 365`	Determines the expiration date included in the certificate. The default lifetime of a local CA certificate is three years. Make sure that you limit the validity period of the certificate to less than the recommended end date of 03:14:08 UTC, January 19, 2038.
Step 3	**no lifetime ca-certificate** **Example:** `hostname (config-ca-server)# no lifetime ca-certificate`	(Optional) Resets the local CA certificate lifetime to the default value of three years. The local CA server automatically generates a replacement CA certificate 30 days before it expires, which allows the replacement certificate to be exported and imported onto any other devices for certificate validation of user certificates that have been issued by the local CA certificate after the current local CA certificate has expired. The following preexpiration syslog message is generated: `%ASA-1-717049: Local CA Server certificate is due to expire in` *days* `days and a replacement certificate is available for export.` **Note** When notified of this automatic rollover, the administrator must make sure that the new local CA certificate is imported onto all required devices before it expires.

Configuring the User Certificate Lifetime

To configure the user certificate lifetime, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**lifetime certificate** *time* **Example:** `hostname (config-ca-server)# lifetime certificate 60`	Sets the length of time that you want user certificates to remain valid. **Note** Before a user certificate expires, the local CA server automatically initiates certificate renewal processing by granting enrollment privileges to the user several days ahead of the certificate expiration date, setting renewal reminders, and delivering an e-mail message that includes the enrollment username and OTP for certificate renewal. Make sure that you limit the validity period of the certificate to less than the recommended end date of 03:14:08 UTC, January 19, 2038.

Configuring the CRL Lifetime

To configure the CRL lifetime, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**lifetime crl** *time* **Example:** `hostname (config-ca-server)# lifetime crl 10`	Sets the length of time that you want the CRL to remain valid. The local CA updates and reissues the CRL each time that a user certificate is revoked or unrevoked, but if no revocation changes occur, the CRL is reissued automatically once each CRL lifetime. If you do not specify a CRL lifetime, the default time period is six hours.

Step 3	**crypto ca server crl issue**
	
	Example:
	`hostname(config)# crypto ca server crl issue`
	`A new CRL has been issued.`
		Forces the issuance of a CRL at any time, which immediately updates and regenerates a current CRL to overwrite the existing CRL.
		
		Note Do not use this command unless the CRL file has been removed in error or has been corrupted and must be regenerated.

Configuring the Server Keysize

To configure the server keysize, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**keysize server** **Example:** `hostname (config-ca-server)# keysize server 2048`	Specifies the size of the public and private keys generated at user-certificate enrollment. The keypair size options are 512, 768, 1024, 2048 bits, and the default value is 1024 bits. **Note** After you have enabled the local CA, you cannot change the local CA keysize, because all issued certificates would be invalidated. To change the local CA keysize, you must delete the current local CA and reconfigure a new one.

Examples

The following is sample output that shows two user certificates in the database.

```
Username: user1
Renewal allowed until: Not Allowed
Number of times user notified: 0
PKCS12 file stored until: 12:45:52 UTC Fri Jan 4 2017
Certificates Issued:
serial:   0x71
issued:   12:45:52 UTC Thu Jan 3 2008
expired:  12:17:37 UTC Sun Dec 31 2017
status:   Not Revoked
Username: user2
Renewal allowed until: Not Allowed
Number of times user notified: 0
PKCS12 file stored until: 12:27:59 UTC Fri Jan 4 2008
Certificates Issued:
serial:   0x2
issued:   12:27:59 UTC Thu Jan 3 2008
expired:  12:17:37 UTC Sun Dec 31 2017
status:   Not Revoked
<--- More --->
```

Setting Up External Local CA File Storage

You can store the local CA server configuration, users, issued certificates, and CRLs in the local CA server database either in flash memory or in an external local CA file system. To configure external local CA file storage, perform the following steps:

	Command	Purpose
Step 1	**mount** *name* **type** **Example:** `hostname (config)# mount mydata type cifs`	Accesses configuration mode for the specific file system type.

Step 2	**mount** *name* **type cifs**
	
	Example:
	`hostname (config-mount-cifs)# mount mydata type cifs`
	`server 10.1.1.10 share myshare`
	`domain example.com`
	`username user6`
	`password ********`
	`status enable`
		Mounts a CIFS file system.
		
		Note Only the user who mounts a file system can unmount it with the **no mount** command.

	Command	Purpose
Step 3	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 4	**database path** *mount-name directory-path* **Example:** `hostname (config-ca-server)# database path mydata:newuser`	Specifies the location of *mydata*, the premounted CIFS file system to be used for the local CA server database. Establishes a path to the server and then specifies the local CA file or folder name to use for storage and retrieval. To return local CA file storage to the ASA flash memory, use the **no database path** command. **Note** To secure stored local CA files on an external server requires a premounted file system of file type CIFS or FTP that is username-protected and password-protected.
Step 5	**write memory** **Example:** `hostname (config)# write memory`	Saves the running configuration. For external local CA file storage, each time that you save the ASA configuration, user information is saved from the ASA to the premounted file system and file location, *mydata:newuser.* For flash memory storage, user information is saved automatically to the default location for the start-up configuration.

Examples

The following example shows the list of local CA files that appear in flash memory or in external storage:

hostname (config-ca-server)# **dir LOCAL* //**

Directory of disk0:/LOCAL*

75 -rwx 32 13:07:49 Jan 20 2007 LOCAL-CA-SERVER.ser
77 -rwx 229 13:07:49 Jan 20 2007 LOCAL-CA-SERVER.cdb
69 -rwx 0 01:09:28 Jan 20 2007 LOCAL-CA-SERVER.udb
81 -rwx 232 19:09:10 Jan 20 2007 LOCAL-CA-SERVER.crl
72 -rwx 1603 01:09:28 Jan 20 2007 LOCAL-CA-SERVER.p12

127119360 bytes total (79693824 bytes free)

Downloading CRLs

To make the CRL available for HTTP download on a given interface or port, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**publish-crl interface** *interface* **port** *portnumber* **Example:** `hostname (config-ca-server)# publish-crl outside 70`	Opens a port on an interface to make the CRL accessible from that interface.The specified interface and port are used to listen for incoming requests for the CRL. The interface and optional port selections are as follows: • inside—Name of interface/GigabitEthernet0/1 • management—Name of interface/ Management0/0 • outside—Name of interface/GigabitEthernet0/0 • Port numbers can range from 1-65535. TCP port 80 is the HTTP default port number. **Note** If you do not specify this command, the CRL is not accessible from the CDP location, because this command is required to open an interface to download the CRL file. The CDP URL can be configured to use the IP address of an interface, and the path of the CDP URL and the filename can also be configured (for example, http://10.10.10.100/user8/my_crl_file). In this case, only the interface with that IP address configured listens for CRL requests, and when a request comes in, the ASA matches the path, /user8/my_crl_file to the configured CDP URL. When the path matches, the ASA returns the stored CRL file. **Note** The protocol must be HTTP, so the prefix displayed is http://.

Storing CRLs

To establish a specific location for the automatically generated CRL of the local CA, perform the following site-to-site task in either single or multiiple context mode:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**cdp-url** *url* **Example:** `hostname(config-ca-server)# cdp-url http://172.16.1.1/pathname/myca.crl`	Specifies the CDP to be included in all issued certificates. If you do not configure a specific location for the CDP, the default URL location is http://*hostname.domain*/+CSCOCA+/asa_ca.crl. The local CA updates and reissues the CRL each time a user certificate is revoked or unrevoked. If no revocation changes occur, the CRL is reissued once each CRL lifetime. If this command is set to serve the CRL directly from the local CA ASA, see the "Downloading CRLs" section on page 1-33 for instructions about opening a port on an interface to make the CRL accessible from that interface. The CRL exists for other devices to validate the revocation of certificates issued by the local CA. In addition, the local CA tracks all issued certificates and status within its own certificate database. Revocation checking is performed when a validating party needs to validate a user certificate by retrieving the revocation status from an external server, which might be the CA that issued the certificate or a server designated by the CA.

Setting Up Enrollment Parameters

To set up enrollment parameters, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**otp expiration** *timeout* **Example:** `hostname(config-ca-server)# otp expiration 24`	Specifies the number of hours that an issued OTP for the local CA enrollment page is valid. The default expiration time is 72 hours. **Note** The user OTP to enroll for a certificate on the enrollment website is also used as the password to unlock the PKCS12 file that includes the issued certificate and keypair for the specified user.
Step 3	**enrollment-retrieval** *timeout* **Example:** `hostname(config-ca-server)# enrollment-retrieval 120`	Specifies the number of hours an already-enrolled user can retrieve a PKCS12 enrollment file.This time period begins when the user is successfully enrolled. The default retrieval period is 24 hours. Valid values for the retrieval period range from 1 to 720 hours. The enrollment retrieval period is independent of the OTP expiration period. After the enrollment retrieval time expires, the user certificate and keypair are no longer available. The only way a user may receive a certificate is for the administrator to reinitialize certificate enrollment and allow a user to log in again.

Adding and Enrolling Users

To add a user who is eligible for enrollment in the local CA database, perform the following commands:

	Command	Purpose
Step 1	**crypto ca server user-db add** *username* [**dn** *dn*] [**email** *emailaddress*] **Example:** `hostname (config-ca-server)# crypto ca server user-db add user1 dn user1@example.com, Engineer, Example Company, US, email user1@example.com`	Adds a new user to the local CA database. Options are as follows: • *username*—A string of 4-64 characters, which is the simple username for the user being added. The username can be an e-mail address, which then is used to contact the user as necessary for enrollment invitations. • *dn*—The distinguished name, a global, authoritative name of an entry in the OSI Directory (X.500) (for example, cn=user1@example.com, cn=Engineer, o=Example Company, c=US). • *e-mail-address*—The e-mail address of the new user to which OTPs and notices are to be sent.
Step 2	**crypto ca server user-db allow** *user* **Example:** `hostname (config-ca-server)# crypto ca server user-db allow user6`	Provides user privileges to a newly added user.
Step 3	**crypto ca server user-db email-otp** *username* **Example:** `hostname (config-ca-server)# crypto ca server user-db email-otp exampleuser1`	Notifies a user in the local CA database to enroll and download a user certificate, which automatically e-mails the OTP to that user. **Note** When an administrator wants to notify a user through e-mail, the administrator must specify the e-mail address in the username field or in the e-mail field when adding that user.

	Command	Purpose
Step 4	**crypto ca server user-db show-otp** **Example:** `hostname (config-ca-server)# crypto ca server user-db show-otp`	Shows the issued OTP.
Step 5	**otp expiration** *timeout* **Example:** `hostname (config-ca-server)# otp expiration 24`	Sets the enrollment time limit in hours. The default expiration time is 72 hours. The **otp expiration** command defines the amount of time that the OTP is valid for user enrollment. This time period begins when the user is allowed to enroll. After a user enrolls successfully within the time limit and with the correct OTP, the local CA server creates a PKCS12 file, which includes a keypair for the user and a user certificate that is based on the public key from the keypair generated and the subject-name DN specified when the user is added. The PKCS12 file contents are protected by a passphrase, the OTP. The OTP can be handled manually, or the local CA can e-mail this file to the user to download after the administrator allows enrollment. The PKCS12 file is saved to temporary storage with the name, *username.p12*. With the PKCS12 file in storage, the user can return within the enrollment-retrieval time period to download the PKCS12 file as many times as needed. When the time period expires, the PKCS12 file is removed from storage automatically and is no longer available to download. **Note** If the enrollment period expires before the user retrieves the PKCS12 file that includes the user certificate, enrollment is not permitted.

Renewing Users

To specify the timing of renewal notices, perform the following steps:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**renewal-reminder** *time* **Example:** `hostname (config-ca-server)# renewal-reminder 7`	Specifies the number of days (1-90) before the local CA certificate expires that an initial reminder to reenroll is sent to certificate owners. If a certificate expires, it becomes invalid. Renewal notices and the times they are e-mailed to users are variable, and can be configured by the administrator during local CA server configuration. Three reminders are sent. An e-mail is automatically sent to the certificate owner for each of the three reminders, provided an e-mail address is specified in the user database. If no e-mail address exists for the user, a syslog message alerts you of the renewal requirement. The ASA automatically grants certificate renewal privileges to any user who holds a valid certificate that is about to expire, as long as the user still exists in the user database. Therefore, if an administrator does not want to allow a user to renew automatically, the administrator must remove the user from the database before the renewal time period.

Restoring Users

To restore a user and a previously revoked certificate that was issued by the local CA server, perform the following steps:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**crypto ca server unrevoke** *cert-serial-no* **Example:** `hostname (config)# crypto ca server unrevoke 782ea09f`	Restores a user and unrevokes a previously revoked certificate that was issued by the local CA server. The local CA maintains a current CRL with serial numbers of all revoked user certificates. This list is available to external devices and can be retrieved directly from the local CA if it is configured to do so with the **cdp-url** command and the **publish-crl** command. When you revoke (or unrevoke) any current certificate by certificate serial number, the CRL automatically reflects these changes.

Removing Users

To delete a user from the user database by username, perform the following steps:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**crypto ca server user-db remove** *username* **Example:** `hostname (config)# crypto ca server user-db remove user1`	Removes a user from the user database and allows revocation of any valid certificates that were issued to that user.

Revoking Certificates

To revoke a user certificate, perform the following steps:

	Command	Purpose
Step 1	**crypto ca server** **Example:** `hostname (config)# crypto ca server`	Enters local ca server configuration mode. Allows you to configure and manage a local CA.
Step 2	**crypto ca server revoke** *cert-serial-no* **Example:** `hostname (config-ca-server)# crypto ca server revoke 782ea09f`	Enters the certificate serial number in hexadecimal format. Marks the certificate as revoked in the certificate database on the local CA server and in the CRL, which is automatically reissued. **Note** The password is also required if the certificate for the ASA needs to be revoked, so make sure that you record it and store it in a safe place.

Maintaining the Local CA Certificate Database

To maintain the local CA certificate database, make sure that you save the certificate database file, LOCAL-CA-SERVER.cdb, with the **write memory** command each time that a change to the database occurs. The local CA certificate database includes the following files:

- The LOCAL-CA-SERVER.p12 file is the archive of the local CA certificate and keypair that is generated when the local CA server is initially enabled.
- The LOCAL-CA-SERVER.crl file is the actual CRL.
- The LOCAL-CA-SERVER.ser file keeps track of the issued certificate serial numbers.

Rolling Over Local CA Certificates

Thirty days before the local CA certificate expires, a rollover replacement certificate is generated, and a syslog message informs the administrator that it is time for local CA rollover. The new local CA certificate must be imported onto all necessary devices before the current certificate expires. If the administrator does not respond by installing the rollover certificate as the new local CA certificate, validations may fail.

The local CA certificate rolls over automatically after expiration using the same keypair. The rollover certificate is available for export in base 64 format.

Examples

The following example shows a base 64 encoded local CA certificate:

```
MIIXlwIBAzCCF1EGCSqGSIb3DQEHAaCCF0IEghc+MIIXOjCCFzYGCSqGSIb3DQEHBqCCFycwghcjAgEAMIIXHAYJKo
ZIhvcNAQcBMBsGCiqGSIb3DQEMAQMwDQQIjph4SxJoyTgCAQGAghbw3v4bFy+GGG2dJnB4OLphsUM+IG3SDOiDwZG9
n1SvtMieoxd7Hxknxbum06JDrujWKtHBIqkrm+td34qlNE1iGeP2YC94/NQ2z+4kS+uZzwcRhl1KEZTS1E4L0fSaC3
uMTxJq2NUHYWmoc8pi4CIeLj3h7VVMy6qbx2AC8I+q57+QG5vG5l5Hi5imwtYfaWwPEdPQxaWZPrzoG1J8BFqdPa1j
BGhAzzuSmElm3j/2dQ3Atro1G9nIsRHgV39fcBgwz4fEabHG7/Vanb+fj81d5nlOiJjDYYbP86tvbZ2yOVZR6aKFVI
0b2AfCr6PbwfC9U8Z/aF3BCyM2sN2xPJrXva94CaYrqyotZdAkSYA5KWScyEcgdqmuBeGDKOncTknfgy0XM+fG5rb3
qAXy1GkjyFI5Bm9Do6RUROoG1DSrQrKeq/hj….
```

```
END OF CERTIFICATE
```

Archiving the Local CA Server Certificate and Keypair

To archive the local CA server certificate and keypair, enter the following command:

Command	Purpose
copy **Example:** `hostname# copy LOCAL-CA-SERVER_0001.pl2 tftp://10.1.1.22/user6/`	Copies the local CA server certificate and keypair and all files from the ASA using either FTP or TFTP. **Note** Make sure that you back up all local CA files as often as possible.

Note

Monitoring Digital Certificates

To display certificate configuration and database information, enter one or more of the following commands:

Command	Purpose
show crypto ca server	Shows local CA configuration and status.
show crypto ca server cert-db	Shows user certificates issued by the local CA.
show crypto ca server certificate	Shows local CA certificates on the console in base 64 format and the rollover certificate when available, including the rollover certificate thumbprint for verification of the new certificate during import onto other devices.
show crypto ca server crl	Shows CRLs.
show crypto ca server user-db	Shows users and their status, which can be used with the following qualifiers to reduce the number of displayed records: • allowed. Shows only users currently allowed to enroll. • enrolled. Shows only users that are enrolled and hold a valid certificate • expired. Shows only users holding expired certificates. • on-hold. Lists only users without a certificate and not currently allowed to enroll.
show crypto ca server user-db allowed	Shows users who are eligible to enroll.
show crypto ca server user-db enrolled	Shows enrolled users with valid certificates.
show crypto ca server user-db expired	Shows users with expired certificates.
show crypto ca server user-db on-hold	Shows users without certificates who are not allowed to enroll.
show crypto key *name of key*	Shows key pairs that you have generated.
show running-config	Shows local CA certificate map rules.

Examples

The following example shows an RSA general-purpose key:

```
hostname/contexta(config)# show crypto key mypubkey
Key pair was generated at: 16:39:47 central Feb 10 2010
Key name: <Default-RSA-Key>
 Usage: General Purpose Key
 Modulus Size (bits): 1024
 Key Data:

  30819f30 0d06092a 864886f7 0d010101 05000381 8d003081 89028181 00ea51b7
  0781848f 78bccac2 4a1b5b8d 2f3e30b4 4cae9f86 f4485207 159108c9 f5e49103
  9eeb0f5d 45fd1811 3b4aafce 292b3b64 b4124a6f 7a777b08 75b88df1 8092a9f8
  5508e9e5 2c271245 7fd1c0c3 3aaf1e04 c7c4efa4 600f4c4a 6afe56ad c1d2c01c
  e08407dd 45d9e36e 8cc0bfef 14f9e6ac eca141e4 276d7358 f7f50d13 79020301 0001
Key pair was generated at: 16:34:54 central Feb 10 2010
```

The following example shows the local CA CRL:

```
hostname (config)# show crypto ca server crl
Certificate Revocation List:
    Issuer: cn=xx5520-1-3-2007-1
    This Update: 13:32:53 UTC Jan 4 2010
    Next Update: 13:32:53 UTC Feb 3 2010
    Number of CRL entries: 2
    CRL size: 270 bytes
Revoked Certificates:
    Serial Number: 0x6f
    Revocation Date: 12:30:01 UTC Jan 4 2010
    Serial Number: 0x47
    Revocation Date: 13:32:48 UTC Jan 4 2010
```

The following example shows one user on-hold:

```
hostname (config)# show crypto ca server user-db on-hold
username: wilma101
email:    <None>
dn:       <None>
allowed:  <not allowed>
notified: 0
hostname (config)#
```

The following example shows output of the **show running-config** command, in which local CA certificate map rules appear:

```
crypto ca certificate map 1
 issuer-name co asc
 subject-name attr ou eq Engineering
```

Feature History for Certificate Management

Table 1-1 lists each feature change and the platform release in which it was implemented.

Table 1-1 Feature History for Certificate Management

Feature Name	Platform Releases	Feature Information
Certificate management	7.0(1)	Digital certificates (including CA certificates, identity certificates, and code signer certificates) provide digital identification for authentication. A digital certificate includes information that identifies a device or user, such as the name, serial number, company, department, or IP address. CAs are trusted authorities that "sign" certificates to verify their authenticity, thereby guaranteeing the identity of the device or user. CAs issue digital certificates in the context of a PKI, which uses public-key or private-key encryption to ensure security.
Certificate management	7.2(1	We introduced the following commands: **issuer-name** *DN-string,* **revocation-check crl none, revocation-check crl, revocation-check none**. We deprecated the following commands: **crl** {**required** | **optional** | **nocheck**}.

Table 1-1 Feature History for Certificate Management (continued)

Feature Name	Platform Releases	Feature Information
Certificate management	8.0(2)	We introduced the following commands: **cdp-url**, **crypto ca server**, **crypto ca server crl issue**, **crypto ca server revoke** *cert-serial-no,* **crypto ca server unrevoke** *cert-serial-no,* **crypto ca server user-db add** *user* [**dn** *dn*] **[email** *e-mail-address*], **crypto ca server user-db allow** {*username* **| all-unenrolled | all-certholders**} [**display-otp**] [**email-otp**] [**replace-otp**], **crypto ca server user-db email-otp** {*username* **| all-unenrolled | all-certholders**}, **crypto ca server user-db remove** *username,* **crypto ca server user-db show-otp** {*username* **| all-certholders | all-unenrolled**}, **crypto ca server user-db write**, **[no] database path** *mount-name directory-path,* **debug crypto ca server** [*level*], **lifetime {ca-certificate | certificate | crl}** *time,* no shutdown, **otp expiration** *timeout,* **renewal-reminder** *time,* **show crypto ca server**, **show crypto ca server cert-db [user** *username* **| allowed | enrolled | expired | on-hold] [serial** *certificate-serial-number*], **show crypto ca server certificate**, **show crypto ca server crl**, **show crypto ca server user-db** [**expired | allowed | on-hold | enrolled**], **show crypto key** *name of key,* **show running-config**, **shutdown**.
SCEP proxy	8.4(1)	We introduced this feature, which provides secure deployment of device certificates from third-party CAs. We introduced the following commands: **crypto ikev2 enable outside client-services port** *portnumber*, **scep-enrollment enable**, **scep-forwarding-url value** *URL*, **secondary-pre-fill-username clientless hide use-common-password** *password*, **secondary-pre-fill-username ssl-client hide use-common-password** *password*, **secondary-username-from-certificate** {**use-entire-name | use-script |** {*primary_attr* [*secondary-attr*]}} [**no-certificate-fallback cisco-secure-desktop machine-unique-id**].