

Digital Certificates

This chapter describes how to configure digital certificates.

- About Digital Certificates, on page 1
- Guidelines for Digital Certificates, on page 8
- Configure Digital Certificates, on page 11
- Set a Certificate Expiration Alert (for Identity or CA Certificates), on page 35
- Monitoring Digital Certificates, on page 35
- History for Certificate Management, on page 37

About Digital Certificates

Digital 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 responsible for managing certificate requests and issuing digital certificates. CAs are trusted authorities that "sign" certificates to verify their authenticity, thereby guaranteeing the identity of the device or user.

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. CAs issue digital certificates in the context of a PKI, which uses public-key or private-key encryption to ensure security.

For authentication using digital certificates, at least one identity certificate and its issuing CA certificate must exist on an ASA. This configuration allows multiple identities, roots, and certificate hierarchies. The ASA evaluates third-party certificates against CRLs, also called authority revocation lists, all the way from the identity certificate up the chain of subordinate certificate authorities.

Descriptions of several different types of available digital certificates follow:

- A CA certificate is used to sign other certificates. It is self-signed and called a root certificate. A certificate that is issued by another CA certificate is called a subordinate certificate.
- CAs also issue identity certificates, which are certificates for specific systems or hosts.
- Code-signer certificates are special certificates that are used to create digital signatures to sign code, with the signed code itself revealing the certificate origin.

In a hierarchy with a root and two intermediate CA certificates, a CRL validation for remote access fail on the headend running 9.13 or later when the ID certificate is signed by one intermediate CA, but the CRL is signed by another intermediate CA. The failure happens even if the headend trusts both the intermediates where both are signed by the same root.

Hence, each signer must maintain their own CRL. Each signer would then specify the location of the CRL in the url list of each certificate it signs. Alternatively, you can configure a url override in the trustpoint of each signer pointing to the correct CRL location.

The local CA integrates an independent certificate authority feature on the ASA, deploys certificates, and provides secure revocation checking of issued certificates. The local CA provides a secure, configurable, in-house authority for certificate authentication with user enrollment through a website login page.

Note CA certificates and identity certificates apply to both site-to-site VPN connections and remote access VPN connections. Procedures in this document refer to remote access VPN use in the ASDM GUI.

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.

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 or Elliptic Curve Signature Algorithm (ECDSA) keys, which have the following characteristics:

- RSA keys can be used for SSH or SSL.
- SCEP enrollment supports the certification of RSA keys.
- The maximum RSA key size is 4096, and the default is 2048.
- The maximum ECDSA key length is 521, and the default is 384.
- 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 the 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.

You can configure Automated Certificate Management Environment (ACME) protocol to ASA trustpoint to manage the TLS device certificates. The ACME-enabled trustpoints also supports manual certificate enrollment. ACME server validates domain ownership using ASA port 80. ACME is supported only in non-clustered single-context ASA deployments.

Proxy for SCEP Requests

The ASA can proxy SCEP requests between Secure Client 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 Secure Client SSL or IKEv2 VPN session. It supports all SCEP-compliant CAs, including Cisco IOS CS, Windows Server 2003 CA, and Windows Server 2008 CA.

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



Note The **revocation-check crl none**, which was removed in 9.13(1), was restored.

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.



Note Though the CRL server responds with HTTP flag "Connection: Keep-alive" to indicate a persistent connection, ASA does not request support for persistent connection. Change the settings on the CRL server to respond with "Connection: Close" when the list is sent.

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.

You could receive a *revocation check* failure for a user connection/certificate if you exceed the CRL size limit of 16 MB.

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. Large CRLs require significant computational overhead to parse them. Hence, for better performance, use many CRLs of smaller size rather than few large CRLs, or preferably, use OCSP.

See the following the cache sizes:

- Single context mode—128 MB
- Multiple context mode—16 MB per context

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.

Note The **revocation-check ocsp none**, which was removed in 9.13(1), was restored.

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 **revocation-check ocsp** command to configure the client certificate.

Certificates and User Login Credentials

The following section describes the different methods of using certificates and user login credentials (username and password) for authentication and authorization. These methods apply to IPsec, Secure Client, and Clientless SSL VPN.

In all cases, LDAP authorization does not use the password as a credential. RADIUS authorization uses either a common password for all users or the username as a password.

User Login Credentials

The default method for authentication and authorization uses the user login credentials.

- Authentication
  - Enabled by the authentication server group setting in the tunnel group (also called ASDM Connection Profile)
  - Uses the username and password as credentials
- Authorization
  - Enabled by the authorization server group setting in the tunnel group (also called ASDM Connection Profile)
  - Uses the username as a credential

Certificates

If user digital certificates are configured, the ASA first validates the certificate. It does not, however, use any of the DNs from certificates as a username for the authentication.

If both authentication and authorization are enabled, the ASA uses the user login credentials for both user authentication and authorization.

- Authentication

- Enabled by the authentication server group setting
- Uses the username and password as credentials

- Authorization
  - Enabled by the authorization server group setting
  - Uses the username as a credential

If authentication is disabled and authorization is enabled, the ASA uses the primary DN field for authorization.

- Authentication
  - DISABLED (set to None) by the authentication server group setting
  - No credentials used

- Authorization
  - Enabled by the authorization server group setting
  - Uses the username value of the certificate primary DN field as a credential

Note If the primary DN field is not present in the certificate, the ASA uses the secondary DN field value as the username for the authorization request.

For example, consider a user certificate that includes the following Subject DN fields and values:

```
Cn=anyuser,OU=sales;O=XYZCorporation;L=boston;S=mass;C=us;ea=anyuser@example.com
```

If the Primary DN = EA (E-mail Address) and the Secondary DN = CN (Common Name), then the username used in the authorization request would be anyuser@example.com.

Guidelines for Digital Certificates

This section includes guidelines and limitations that you should check before configuring digital certificates.

Context Mode Guidelines

- Supported in single context mode only for third-party CAs.

Failover Guidelines

- Does not support replicating sessions in Stateful Failover.
- Does not support failover for local CAs.
- Certificates are automatically copied to the standby unit if you configure stateful failover. If you find a certificate is missing, use the **write standby** command on the active unit.

IPv6 Guidelines

Supports IPv6 OCSP and CRL URLs. You must enclose IPv6 addresses in square brackets, for example: *http://[0:0:0:0:0.18:0a01:7c16]*.

Local CA Certificates

- 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.
- 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.
- 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.

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

```
MIIXlwIBAzCCF1EGCSqGSIb3DQEHAaCCF0IEghc+MIIXOjCCFzYGCSqGSIb3DQEHBqCCFycwghcjAgEAMIIXHA
YJKoZIhvcNAQcBMBsGCiqGSIb3DQEMAQMwDQQIjph4SxJoyTgCAQGAghbw3v4bFy+GGG2dJnB4OLphsUM+IG3S
DOiDwZG9n1SvtMieoxd7Hxknxbum06JDrujWKtHBIqkrm+td34qlNE1iGeP2YC94/NQ2z+4kS+uZzwcRhl1KEZ
TS1E4L0fSaC3uMTxJq2NUHYWmoc8pi4CIeLj3h7VVMy6qbx2AC8I+q57+QG5vG5l5Hi5imwtYfaWwPEdPQxaWZ
PrzoG1J8BFqdPa1jBGhAzzuSmElm3j/2dQ3Atro1G9nIsRHgV39fcBgwz4fEabHG7/Vanb+fj81d5nlOiJjDYY
bP86tvbZ2yOVZR6aKFVI0b2AfCr6PbwfC9U8Z/aF3BCyM2sN2xPJrXva94CaYrqyotZdAkSYA5KWScyEcgdqmu
BeGDKOncTknfgy0XM+fG5rb3qAXy1GkjyFI5Bm9Do6RUROoG1DSrQrKeq/hj….

END OF CERTIFICATE
```

SCEP Proxy Support

- Ensure that the ASA and the Cisco ISE Policy Service Nodes are synchronized using the same NTP server.
- Secure 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.

Additional Guidelines

- The type of certificate you can use is constrained by the certificate types supported by the applications that will use the certificate. RSA certificates are generally supported by all applications that use certificates. But EDDSA certificates might not be supported by workstation operating systems, browsers, ASDM, or Secure Client. For example, you need to use an RSA certificate for remote access VPN identity and authentication. For site-to-site VPN, where the ASA is the application that uses the certificate, EDDSA is supported.
- 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.
- The ASA establishes LDAP/SSL connection only if one of the following certification criteria is satisfied:
  - The LDAP server certificate is trusted (exists in a trustpoint or the ASA trustpool) and is valid.
  - A CA certificate from servers issuing chain is trusted (exists in a trustpoint or the ASA trustpool) and all subordinate CA certificates in the chain are complete and valid.
- 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. We recommend using a key size of at least 2048.
- 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 Secure Client 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 certificate parameters when you import them on the ASA. In ASA, these certificates are decoded to build them into internal data structures. Certificates with blank fields are construed as non-compliant with the decoding standards, and hence the installation validation fails. However, from version 9.16, blank values of optional fields does not impact decoding and installation validation criteria.
- 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
```

- Configure DNS server on the device for a sucessful ACME enrollment. For information on DNS server configuration procedure, see Configure the DNS Servers.

Configure Digital Certificates

The following topics explain how to configure digital certificates.

Configure Key Pairs

To create or remove key pairs, perform the following steps.

Procedure

Step 1 Generate one default, general-purpose RSA key pair.

crypto key generate rsa modulus 2048

Example:

```
ciscoasa(config)# crypto key generate rsa modulus 2048
```

The default key modulus is 2048, but you should specify the modulus explicitly to ensure you get the size you require. The key is named Default-RSA-Key.

For RSA keys, the modulus can be one of the following (in bits): 2048 or 4096.

If you also want an Elliptic Curve Signature Algorithm (ECDSA) key, you can generate the Default-ECDSA-Key. The default length is 384, but you can also use 256 or 521.

crypto key generate ecdsa elliptic-curve 384

If you also want an Edwards Curve Signature Algorithm (EdDSA) key, you can generate the Default-EdDSA-Key. The default length is 256 bits.

Note
EST enrollments on the ASA using keypairs of type EdDSA (Ed25519) is not supported. EST enrollments can use only RSA or ECDSA keys.

crypto key generate eddsa edward-curve Ed25519

Step 2 (Optional) Create additional keys with unique names.

crypto key generate rsa label *key-pair-label* **modulus** *size*

crypto key generate ecdsa label *key-pair-label* **elliptic-curve** *size*

Example:

```
ciscoasa(config)# crypto key generate rsa label exchange modulus 2048
```

The label is referenced by the trustpoint that uses the key pair.

Step 3 Verify key pairs that you have generated.

show crypto key mypubkey {**rsa** | **ecdsa**}

Example:

```
ciscoasa/contexta(config)# show crypto mypubkey key rsa
```

Step 4 Save the key pair that you have generated.

write memory

Example:

```
ciscoasa(config)# write memory
```

Step 5 If necessary, remove existing key pairs so that you can generate new ones.

crypto key zeroize {**rsa** | **ecdsa**}

Example:

```
ciscoasa(config)# crypto key zeroize rsa
```

Step 6 (Optional) Archive the local CA server certificate and key pair.

copy

Example:

```
ciscoasa# copy LOCAL-CA-SERVER_0001.p12 tftp://10.1.1.22/user6/
```

This command copies the local CA server certificate and key pair 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.

Step 7 For enrollment methods that support auto-enrollment, key pairs are automatically generated and stored in flash at the time of enrollment. To enable this support, following key parameters are specified in the trustpoint:

The following example shows ECDSA keypair configuration:

```
ciscoasa(config-ca-trustpoint)# keypair ?

crypto-ca-trustpoint mode commands/options:
  WORD < 129 char  Name of key pair
  ecdsa            Generate ECDSA keys
  eddsa            Generate EDDSA keys
  rsa              Generate RSA keys
ciscoasa(config-ca-trustpoint)# keypair ecdsa elliptic-curve ?
```

```
crypto-ca-trustpoint mode commands/options:
  256  256 bits
  384  384 bits
  521  521 bits
ciscoasa(config-ca-trustpoint)# keypair ecdsa elliptic-curve 521
ciscoasa(config-ca-trustpoint)#
```

The following example shows RSA keypair configuration:

```
ciscoasa(config-ca-trustpoint)# keypair ?

crypto-ca-trustpoint mode commands/options:
  WORD < 129 char  Name of key pair
  ecdsa            Generate ECDSA keys
  eddsa            Generate EDDSA keys
  rsa              Generate RSA keys
ciscoasa(config-ca-trustpoint)# keypair ecdsa elliptic-curve ?

ciscoasa(config-ca-trustpoint)# keypair rsa modulus ?

crypto-ca-trustpoint mode commands/options:
  2048  2048 bits
  3072  3072 bits
  4096  4096 bits
ciscoasa(config-ca-trustpoint)# keypair rsa modulus 2048
ciscoasa(config-ca-trustpoint)#
```

Note
EDDSA keys are not supported for ACME. CLI errors occur when EDDSA key pair selection is already present when ACME enrollment protocol is added to the truspoint and when you attempt to add EDDSA key pair to the trustpoint that is configured for ACME.

Step 8 You can specify whether the enrollment request will use the the same key pair that is currently in use or generate a new one by running the following command:

crypto ca enroll my_acme_tp regenerate

The following example shows configuration of enrollment for a trustpoint to retain the same key pair that is currently in use:

Example:

```
ciscoasa(config)# crypto ca enroll my_acme_tp ?
configure mode commands/options:
  noconfirm   Specify this keyword to suppress all interactive prompting.
  regenerate  Regenerate the key pair to be used for enrollment
  <cr>
ciscoasa(config)# crypto ca enroll my_acme_tp noconfirm
```

Step 9 You can remove the key pairs by running the following command:

crypto key zeroize rsa

The following example shows how to remove key pairs:

Example:

```
ciscoasa(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
```

Configure Trustpoints

To configure a trustpoint, perform the following steps:

Procedure

Step 1 Create a trustpoint that corresponds to the CA from which the ASA needs to receive a certificate.

crypto ca trustpoint *trustpoint-name*

Example:

```
ciscoasa/contexta(config)# crypto ca trustpoint Main
```

You enter the crypto ca trustpoint configuration mode, which controls CA-specific trustpoint parameters that you may configure starting in Step 3.

Step 2 Specify the source interface for the CA certification:

enrollment interface *interface-name*

```
ciscoasa(config-ca-trustpoint)#  enrollment interface mgmt
```

Step 3 Choose one of the following options:

- Request automatic enrollment using ACME with the specified trustpoint:

  a. Configure the enrollment URL by running the following command:

  enrollment protocol acme *url*

  Example:

```
ciscoasa(config-ca-trustpoint)# enrollment protocol acme ?
crypto-ca-trustpoint mode commands/options:
  authentication  ACME authentication method
  url             CA server enrollment URL
ciscoasa(config-ca-trustpoint)# enrollment protocol acme url ?
crypto-ca-trustpoint mode commands/options:
  LINE < 477 char  URL
  LetsEncrypt      Use the Let's Encrypt CA

ciscoasa(config-ca-trustpoint)#  enrollment protocol acme url
https://mytest.com:8443/acme/acme/directory
```

  b. Configure the authentication method and interface for which a certificate is requested by running the following command:

  enrollment protocol acme authentication http01 *inf-name*

```
ciscoasa(config-ca-trustpoint)# enrollment protocol acme ?
crypto-ca-trustpoint mode commands/options:
  authentication  ACME authentication method
  url             CA server enrollment URL

ciscoasa(config-ca-trustpoint)# enrollment protocol acme authentication ?
crypto-ca-trustpoint mode commands/options:
  http01  Use the HTTP-01 method, which opens port 80 on the specified interface

ciscoasa(config-ca-trustpoint)#  enrollment protocol acme authentication http01 mgmt
```

- Request automatic enrollment using SCEP with the specified trustpoint and configure the enrollment URL by running the following command:

  enrollment protocol scep *url*

  Example:

  ```
  ciscoasa/contexta(config-ca-trustpoint)# enrollment protocol scep url
  http://10.29.67.142:80/certsrv/mscep/mscep.dll
  ```

- Request automatic enrollment using CMP with the specified trustpoint and configure the enrollment URL by running the following command:

  enrollment protocol cmp *url*

  Example

  ```
  ciscoasa/ contexta(config-ca-trustpoint)# enrollment protocol cmp url
  http://10.29.67.142:80/certsrv/mscep/mscep.dll
  ```

  Note
  Carrier license must be included to enable CMP enrollment.

- Request manual enrollment with the specified trustpoint by pasting the certificate received from the CA to the terminal by running the following command:

  enrollment terminal

  ```
  ciscoasa/contexta(config-ca-trustpoint)# enrollment terminal
  ```

- Request self signed certificate by running the following command:

  enrollment self

- Request automatic enrollment using EST with the specified trustpoint and configure the enrollment URL by running the following command:

  enrollment protocol est *url*

  Example

  ```
  asa(config-ca-trustpoint)# enrollment protocol est ?

  crypto-ca-trustpoint mode commands/options:
    url CA server enrollment URL
  asa(config-ca-trustpoint)# enrollment protocol est url ?
   crypto-ca-trustpoint mode commands/options:
  ```

```
  LINE < 477 char  URL
asa(config-ca-trustpoint)# enrollment protocol est url https://xyz.com/est
```

Step 4 If the trustpoint has been configured to use CMP or ACME in the step above, you can optionally enable the functionality that automatically requests certificates. Enter a percentage of the absolute lifetime of the certificate after which auto-enroll will be necessary and specify if you want to generate a new key while renewing the certificate.

```
[no] auto-enroll [<percent>] [regenerate]
```

Step 5 Specify the available CRL configuration options.

revocation-check crl none

Note
The **revocation-check crl none**, which was removed in 9.13(1), was restored.

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# revocation-check crl
ciscoasa/contexta(config-ca-trustpoint)# revocation-check none
```

Note
To enable either required or optional CRL checking, make sure that you configure the trustpoint for CRL management after obtaining certificates.

Step 6 Enable or disable the basic constraints extension and CA flag.

[**no**] **ca-check**

The basic constraints extension identifies whether the subject of the certificate is a Certificate Authority (CA), in which case the certificate can be used to sign other certificates. The CA flag is part of this extension. The presence of these items in a certificate indicates that the certificate's public key can be used to validate certificate signatures.

The **ca-check** command is enabled by default, so you need to enter this command only if you want to disable basic constraints and the CA flag.

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# no ca-check
```

Step 7 (Not applicable for ACME) During enrollment, ask the CA to include the specified e-mail address in the Subject Alternative Name extension of the certificate.

email *address*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# email example.com
```

Step 8 (Optional) Specify a retry period in minutes, and applies only to SCEP enrollment.

enrollment retry period

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# enrollment retry period 5
```

Step 9 (Optional) Specify a maximum number of permitted retries, and applies only to SCEP enrollment.

enrollment retry count

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# enrollment retry period 2
```

Step 10 During enrollment, ask the CA to include the specified fully qualified domain name in the Subject Alternative Name extension of the certificate.

fqdn *fqdn*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# fqdn example.com
```

Step 11 During enrollment, ask the CA to include multiple fully qualified domain name values in the Subject Alternative Name extension of the certificate or requests for manual, SCEP, CMP, EST, ACME, and self-signed certificates. The FQDN values can have a maximum length of 128 characters.

alt-fqdn *fqdn*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# fqdn primary example.com
ciscoasa/contexta(config-ca-trustpoint)# alt-fqdn example1.com
ciscoasa/contexta(config-ca-trustpoint)# alt-fqdn example2.com
ciscoasa/contexta(config-ca-trustpoint)# alt-fqdn example3.com
```

Step 12 (Not recommended for ACME) During enrollment, ask the CA to include the IP address of the ASA in the certificate.

ip-address *ip-address*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# ip-address 10.10.100.1
```

Step 13 Specify the key pair whose public key is to be certified.

keypair *name*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# keypair exchange
```

Step 14 If you have trustpoints configured for CMP or ACME, determine if you want to generate EDDSA keys, EDCSA keys or RSA keys for any manual and automatic enrollments.

```
no keypair name | [rsa modulus 2048|4096] | [edcsa elliptic-curve 256|384|521] | [ eddsa
edwards-curve Ed25519 ]
```

Note
EST enrollments on the ASA using keypairs of type EDDSA (Ed25519) is not supported. EST enrollments can use only RSA and ECDSA keys. ACME enrollment with keypair type EDDSA is not supported.

Note
When the ECDHE_ECDSA cipher group is used, configure the trustpoint with a certificate that contain an ECDSA-capable key. Certificates with RSA key are not compatible with ECDSA ciphers.

Step 15 Configure OCSP URL overrides and trustpoints to use for validating OCSP responder certificates.

match certificate map-name override ocsp

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# match certificate examplemap override ocsp
```

Step 16 Configure the source interface for ASA to reach OCSP:

interface *nameif*

Example:

```
ciscoasa(config)# crypto ca trustpoint TP
ciscoasa(config-ca-trustpoint)# ocsp ?

crypto-ca-trustpoint mode commands/options:
  disable-nonce  Disable OCSP Nonce Extension
  interface      Configure Source interface
  url            OCSP server URL
ciscoasa(config-ca-trustpoint)# ocsp  interface
ciscoasa(config-ca-trustpoint)# ocsp interface ?

crypto-ca-trustpoint mode commands/options:
Current available interface(s):
  inside   Name of interface GigabitEthernet0/0.100
  inside1  Name of interface GigabitEthernet0/0.41
  mgmt     Name of interface Management0/0
  outside  Name of interface GigabitEthernet0/0.51
ciscoasa(config-ca-trustpoint)# ocsp interface mgmt
```

Step 17 Disable the nonce extension on an OCSP request. The nonce extension cryptographically binds requests with responses to avoid replay attacks.

ocsp disable-nonce

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# ocsp disable-nonce
```

Step 18 Configure 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.

ocsp url

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# ocsp url
```

ASA supports both IPv4 and IPv6 OCSP urls. Enclose IPv6 addresses in square brackets, for example: *http://[0:0:0:0:0.18:0a01:7c16]*.

Step 19 Specify a challenge phrase that is registered with the CA during enrollment. The CA usually uses this phrase to authenticate a subsequent revocation request.

password *string*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# password mypassword
```

Step 20 Set one or more methods for revocation checking: CRL, OCSP, and none.

Note
When you are assigning OCSP URL for revocation checking, you can specify the interface (includes management interface) from where the OCSP is reachable. This interface value determines the routing decision.

revocation check

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# revocation check
```

Step 21 During enrollment, ask 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.”).

subject-name *X.500 name*

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# myname X.500 examplename
```

Step 22 During enrollment, ask the CA to include the ASA serial number in the certificate.

serial-number

Example:

```
ciscoasa/contexta(config-ca-trustpoint)# serial number JMX1213L2A7
```

Step 23 Save the running configuration.

write memory

Example:

```
ciscoasa/contexta(config)# write memory
```

Configure 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:

Procedure

Step 1 Enter crypto ca trustpoint configuration mode for the trustpoint whose CRL configuration you want to modify.

crypto ca trustpoint *trustpoint-name*

Example:

```
ciscoasa (config)# crypto ca trustpoint Main
```

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 Enter crl configuration mode for the current trustpoint.

crl configure

Example:

```
ciscoasa(config-ca-trustpoint)# crl configure
```

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 Choose one of the following to configure retrieval policy:

Note
ASA supports IPv4 or IPv6 based CDP and Static URLs. Enclose IPv6 addresses in square brackets, for example: *http://[0:0:0:0:0.18:0a01:7c16]*.

- CRLs are retrieved only from the CRL distribution points (CDP) URLs that are specified in authenticated certificates.

  policy cdp

  ```
  ciscoasa(config-ca-crl)# policy cdp
  ```

  Note
  SCEP retrieval is not supported by distribution points specified in certificates.

- CRLs are retrieved only from certificate map match rule that you configure.

  policy static

```
ciscoasa(config-ca-crl)# policy static
```

- CRLs are retrieved from CRL distribution points specified in authenticated certificates and from certificate map match rule that you configure.

  policy both

```
ciscoasa(config-ca-crl)# policy both
```

Step 4 If you used the **static** or **both** keywords when you configured the CRL policy, you must configure certificate map match rule for CRL retrieval. You can now configure multiple static CDPs to a single map.

enrollment terminal

To remove a specific instance, in the **no** form of the command, include the sequence number or the URL. Ensure the specified values match the configured values. To remove all the entries of the map, simply use the **no** command.

Example:

```
ciscoasa(crypto ca trustpoint)#enrollment terminal

ciscoasa(crypto ca trustpoint)#match certificate Main override cdp  10 url http://192.0.2.10
ciscoasa(crypto ca trustpoint)#match certificate Main override cdp  20 url http://192.0.2.12
ciscoasa(crypto ca trustpoint)#match certificate Main override cdp  30 url http://192.0.2.13
```

Step 5 Specify HTTP, LDAP, or SCEP as the CRL retrieval method.

protocol http | **ldap** | **scep**

Example:

```
ciscoasa(config-ca-crl)# protocol http
```

Step 6 Configure how long the ASA caches CRLs for the current trustpoint. The *refresh-time* argument is the number of minutes that the ASA waits before considering a CRL stale.

cache-time *refresh-time*

Example:

```
ciscoasa(config-ca-crl)# cache-time 420
```

Step 7 Choose one of the following:

- Require the NextUpdate field to be present in CRLs. This is the default setting.

  enforcenextupdate

```
ciscoasa(config-ca-crl)# enforcenextupdate
```

- Allow the NextUpdate field to be absent in CRLs.

no enforcenextupdate

```
ciscoasa(config-ca-crl)# no enforcenextupdate
```

Step 8 Identify 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.

ldap-defaults *server*

Example:

```
ciscoasa (config-ca-crl)# ldap-defaults ldap1
```

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 Allow CRL retrieval if the LDAP server requires credentials.

ldap-dn *admin-DN password*

Example:

```
ciscoasa (config-ca-crl)# ldap-dn cn=admin,ou=devtest,o=engineering c00lRunZ
```

Step 10 Retrieve the current CRL from the CA represented by the specified trustpoint and test the CRL configuration for the current trustpoint.

crypto ca crl request *trustpoint*

Example:

```
ciscoasa (config-ca-crl)# crypto ca crl request Main
```

Step 11 Save the running configuration.

write memory

Example:

```
ciscoasa (config)# write memory
```

Export or Import a Trustpoint Configuration

To export and import a trustpoint configuration, perform the following steps:

Procedure

Step 1 Export a trustpoint configuration with all associated keys and certificates in PKCS12 format.

crypto ca export *trustpoint*

Example:

```
ciscoasa(config)# crypto ca export Main
```

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.

Step 2 Import keypairs and issued certificates that are associated with a trustpoint configuration.

crypto ca import *trustpoint* **pkcs12***passphrase*

Example:

```
ciscoasa(config)# crypto ca import Main pkcs12 ?
```

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 exports PKCS12 data for the trustpoint Main with the passphrase Wh0zits:

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

Exported pkcs12 follows:

[ PKCS12 data omitted ]

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

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

```
ciscoasa (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:

```
ciscoasa (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
```

Configure 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.

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

Procedure

Step 1 Enter CA certificate map configuration mode for the rule you want to configure and specify the rule sequence number.

crypto ca certificate map [*map_name*]*sequence-number*

Example:

```
ciscoasa(config)# crypto ca certificate map test-map 10
```

If you do not specify the map name, the rule is added to the default map: DefaultCertificateMap. For each rule number, you can specify one or more fields to match.

Step 2 Specify the issuer name or subject name:

{**issuer-name** | **subject-name**} [**attr** *attribute*] *operator string*

Example:

```
ciscoasa(config-ca-cert-map)# issuer-name cn=asa.example.com
ciscoasa(config-ca-cert-map)# subject-name attr cn eq mycert
ciscoasa(config-ca-cert-map)# subject-name attr uid eq jcrichton
```

You can match the entire value, or specify the attributes that you want to match. The following are valid attributes:

- c—Country
- cn—Common Name
- dc—Domain Component

- dnq—DN Qualifier
- emailAddress—Email Address
- genq—Generational Qualifier
- gn—Given Name
- i—Initials
- ip—IP Address
- l—Locality
- n—Name
- o—Organization Name
- ou—Organizational Unit
- ser—Serial Number

  Note
  Ensure you specify the serial number attribute in the subject-name. The certificate map only matches with a serial number attribute specified in the subject-name.

- sn—Surname
- sp—State/Province
- t—Title
- uid—User ID
- uname—Unstructured Name

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 3 Specify the alternative subject name:

alt-subject-name *operator string*

Example:

```
ciscoasa(config-ca-cert-map)# alt-subject-name eq happydays
```

The following are valid operators:

- eq—The field must be identical to the value given.
- ne—The field cannot be identical to the value given.

- co—Part or all of the field must match the value given.
- nc—No part of the field can match the value given.

Step 4 Specify the extended key usage:

extended-key-usage *operator OID_string*

Example:

```
ciscoasa(config-ca-cert-map)# extended-key-usage nc clientauth
```

The following are valid operators:

- co—Part or all of the field must match the value given.
- nc—No part of the field can match the value given.

The following are valid OID strings:

- *string*—User-defined string.
- clientauth—Client Authentication (1.3.6.1.5.5.7.3.2)
- codesigning—Code Signing (1.3.6.1.5.5.7.3.3)
- emailprotection—Secure Email Protection (1.3.6.1.5.5.7.3.4)
- ocspsigning—OCSP Signing (1.3.6.1.5.5.7.3.9)
- serverauth—Server Authentication (1.3.6.1.5.5.7.3.1)
- timestamping—Time Stamping (1.3.6.1.5.5.7.3.8)

Configure Reference Identities

When the ASA is acting as a TLS client, it supports rules for verification of an application server's identity as defined in RFC 6125. This RFC specifies procedures for representing the reference identities (configured on the ASA) and verifying them against the presented identities (sent from the application server). If the presented identity cannot be matched against the configured reference identity, the connection is not established and an error is logged.

The server presents its identity by including one or more identifiers in the server certificate presented to the ASA while establishing the connection. Reference identities are configured on the ASA, to be compared to the identity presented in a server certificate during connection establishment. These identifiers are specific instances of the four identifier types specified in RFC 6125. The four identifier types are:

- CN_ID: A Relative Distinguished Name (RDN) in a certificate subject field that contains only one attribute-type-and-value pair of type Common Name (CN), where the value matches the overall form of a domain name. The CN value cannot be free text. A CN-ID reference identifier does not identify an application service.
- DNS-ID: A subjectAltName entry of type dNSName. This is a DNS domain name. A DNS-ID reference identifier does not identify an application service.

- SRV-ID: A subjectAltName entry of type otherName whose name form is SRVName as defined in RFC 4985. A SRV-ID identifier may contain both a domain name and an application service type. For example, a SRV-ID of "_imaps.example.net" would be split into a DNS domain name portion of "example.net" and an application service type portion of "imaps."
- URI-ID: A subjectAltName entry of type uniformResourceIdentifier whose value includes both (i) a "scheme" and (ii) a "host" component (or its equivalent) that matches the "reg-name" rule specified in RFC 3986. A URI-ID identifier must contain the DNS domain name, not the IP address, and not just the hostname. For example, a URI-ID of "sip:voice.example.edu" would be split into a DNS domain name portion of "voice.example.edu" and an application service type of "sip."

A reference identity is created when configuring one with a previously unused name. Once a reference identity has been created, the four identifier types and their associated values can be added or deleted from the reference identity. The reference identifiers MAY contain information identifying the application service and MUST contain information identifying the DNS domain name.

Before you begin

- Reference identities are used when connecting to the Syslog Server and the Smart Licensing server only. No other ASA SSL client mode connections currently support the configuration or use of reference identities.
- ASA implements all the rules for matching the identifiers described in RFC 6125 except for pinned certificates and fallback for interactive clients.
- Ability to pin certificates is not implemented. Therefore, `No Match Found, Pinned Certificate` will not occur. Also, a user will not be given the opportunity to pin a certificate if a match is not found since our implementation is not an interactive client.

Procedure

Step 1 Enter the **[no] crypto ca reference-identity** command in global configuration mode to place the ASA in ca-reference-identity mode.

[no] crypto ca reference-identity *reference-identity-name*

If a reference identity with this *reference-identity-name* is not found, a new reference identity is created. If the **no** form of the command is issued for a reference identity that is still in use, a warning is displayed and the reference identity is not deleted.

Step 2 Enter reference-ids while in ca-reference-identity mode. Multiple reference-ids of any type may be added to the reference identity.

- **[no] cn-id** *value*
- **[no] dns-id** *value*
- **[no] srv-id** *value*
- **[no] uri-id** *value*

To remove a reference identity, use the no form of the command.

Example

Configure a reference identity for RFC 6125 server certificate validation for a syslog server:

```
ciscoasa(config)# crypto ca reference-identity syslogServer
ciscoasa(config-ca-ref-identity)# dns-id syslog1-bxb.cisco.com
ciscoasa(config-ca-ref-identity)# cn-id syslog1-bxb.cisco.com
```

What to do next

Use the reference identity when configuring the Syslog and the Smart Call Home server connections.

Obtain Certificates Manually

To obtain certificates manually, perform the following steps:

Before you begin

You must have already obtained a base-64 encoded CA certificate from the CA represented by the trustpoint.

Procedure

Step 1 Import the CA certificate for the configured trustpoint.

crypto ca authenticate *trustpoint*

Example:

```
ciscoasa(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+JbRTANBgkqhkiG9w0BAQUFADCB
[ 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
```

Whether a trustpoint requires that you manually obtain certificates is determined by the use of the **enrollment terminal** command when you configure the trustpoint.

Step 2 Enroll the ASA with the trustpoint.

crypto ca enroll *trustpoint*

Example:

```
ciscoasa(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:

MIIBoDCCAQkCAQAwIzEhMB8GCSqGSIb3DQEJAhYSRmVyYWxQaXguY2lzY28uY29t
[ certificate request data omitted ]
jF4waw68eOxQxVmdgMWeQ+RbIOYmvt8g6hnBTrd0GdqjjVLt

---End - This line not part of the certificate request---

Redisplay enrollment request? [yes/no]: n
```

This command 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.

Step 3 When a trustpoint is configured for CMP, either a shared secret value (ir) or the name of the trustpoint that contains the certificate that will sign the request (cr) can be specified, but not both. Provide either an out-of-band value by the CA that is used to confirm the authenticity and integrity of messages exchanged with ASA or provide the name of the trustpoint with a previously-issued device certificate used for signing the CMP enrollment request. The shared-secret or signing-certificate keywords are only available when the trustpoint enrollment protocol is set to CMP.

```
crypto ca enroll trustpoint [regenerate] [shared-secret <value> | signing-certificate <value>
```

Step 4 Determine whether or not a new keypair should be generated prior to building the enrollment request.

```
crypto ca enroll trustpoint [regenerate] [shared-secret <value> | signing-certificate <value>
```

Step 5 Import each certificate you receive from the CA and make sure that you paste the certificate to the terminal in base-64 format.

crypto ca import *trustpoint* **certificate**

Example:

```
ciscoasa (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
```

Step 6 Verify that the enrollment process was successful by displaying certificate details issued for the ASA and the CA certificate for the trustpoint.

show crypto ca certificate

Example:

```
ciscoasa(config)# show crypto ca  certificate Main
```

Step 7 Save the running configuration.

write memory

Example:

```
ciscoasa(config)# write memory
```

Step 8 Repeat these steps for each trustpoint that you configure for manual enrollment.

Obtain Certificates Automatically with SCEP

This section describes how to obtain certificates automatically using SCEP.

Before you begin

You must have already obtained a base-64 encoded CA certificate from the CA represented by the trustpoint.

Procedure

Step 1 Obtain the CA certificate for the configured trustpoint.

crypto ca authenticate *trustpoint*

Example:

```
ciscoasa/contexta(config)# crypto ca authenticate Main
```

When you configure the trustpoint, use of the **enrollment url** command determines whether or not you must obtain certificates automatically via SCEP.

Step 2 Enroll the ASA with the trustpoint. This command 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.

crypto ca enroll *trustpoint*

Example:

```
ciscoasa/contexta(config)# crypto ca enroll Main
```

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.

Step 3 Verify that the enrollment process was successful by displaying certificate details issued for the ASA and the CA certificate for the trustpoint.

show crypto ca certificate

Example:

```
ciscoasa/contexta(config)# show crypto ca certificate Main
```

Step 4 Save the running configuration.

write memory

Example:

```
ciscoasa/contexta(config)# write memory
```

Configure Proxy Support for SCEP Requests

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

Procedure

Step 1 Enter tunnel-group ipsec-attributes configuration mode.

tunnel-group *name* **ipsec-attributes**

Example:

```
ciscoasa(config)# tunnel-group remotegrp ipsec-attributes
```

Step 2 Enable client services.

crypto ikev2 enable outside client-services port *portnumber*

Example:

```
ciscoasa(config-tunnel-ipsec)# crypto ikev2 enable outside client-services
```

The default port number is 443.

Note
This command is needed only if you support IKEv2.

Step 3 Enter tunnel-group general-attributes configuration mode.

tunnel-group *name* **general-attributes**

Example:

```
ciscoasa(config)# tunnel-group 209.165.200.225 general-attributes
```

Step 4 Enable SCEP enrollment for the tunnel group.

scep-enrollment enable

Example:

```
ciscoasa(config-tunnel-general)# scep-enrollment enable
INFO: 'authentication aaa certificate' must be configured to complete setup of this option.
```

Step 5 Enter group-policy attributes configuration mode.

group-policy *name* **attributes**

Example:

```
ciscoasa(config)# group-policy FirstGroup attributes
```

Step 6 Enroll the SCEP CA for the group policy. Enter this command once per group policy to support a third-party digital certificate.

scep-forwarding-url value *URL*

Example:

```
ciscoasa(config-group-policy)# scep-forwarding-url value http://ca.example.com:80/
```

URL is the SCEP URL on the CA.

Step 7 Supply a common, secondary password when a certificate is unavailable for WebLaunch support of the SCEP proxy.

secondary-pre-fill-username clientless hide use-common-password *password*

Example:

```
ciscoasa(config)# tunnel-group remotegrp webvpn-attributes
ciscoasa(config-tunnel-webvpn)# secondary-pre-fill-username clientless hide
use-common-password secret
```

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, Secure Client disconnects, then reconnects to the ASA to qualify for a DAP policy that provides access to internal network resources.

Step 8 Hide the secondary prefill username for Secure Client VPN sessions.

secondary-pre-fill-username ssl-client hide use-common-password *password*

Example:

```
ciscoasa(config-tunnel-webvpn)# secondary-pre-fill-username ssl-client hide
use-common-password secret
```

Despite the **ssl-client** keyword inherited from earlier releases, use this command to support Secure Client sessions that use either IKEv2 or SSL.

You must use the **hide** keyword to support the SCEP proxy.

Step 9 Supply the username when a certificate is unavailable.

secondary-username-from-certificate {**use-entire-name** | **use-script** | {*primary_attr* [*secondary-attr*]}} [**no-certificate-fallback cisco-secure-desktop machine-unique-id**]

Example:

```
ciscoasa(config-tunnel-webvpn)# secondary-username-from-certificate CN no-certificate-fallback
 cisco-secure-desktop machine-unique-id
```

Configure CAs

-

Configure Auto-Import of Trustpool Certificates

Smart licensing uses the Smart Call Home infrastructure. When the ASA configures Smart Call Home anonymous reporting in the background, the ASA automatically creates a trustpoint containing the certificate of the CA that issued the Call Home server certificate. The ASA now supports validation of the certificate if the issuing hierarchy of the server certificate changes, without the need for customer involvement to adjust certificate hierarchy changes. You can automate the update of the trustpool bundle at periodic intervals so that Smart Call Home can remain active if the self-signed certificate of the CA server changes. This feature is not supported under multi-context deployments.

Procedure

	Command or Action	Purpose
Step 1	Automatic import of trustpool certificate bundles requires you to specify the URL that ASA uses to download and import the bundle. Use the following command so the import	`ciscoasa(config-ca-trustpool)# auto-import-url Default`

	Command or Action	Purpose
	happens daily at a regular interval with the default Cisco URL and default time of 22 hours:	
Step 2	You can also enable auto import with a custom URL with the following command:	`ciscoasa(config-ca-trustpool)# auto-import url http://www.thawte.com`
Step 3	To give you more flexibility to set downloads during off peak hours or other convenient times, enter the following command which enables the import with a custom time:	`ciscoasa(config-ca-trustpool)# auto-import time 23:23:23`
Step 4	Setting the automatic import with both a custom URL and custom time requires the following command:	`ciscoasa(config-ca-trustpool)# auto-import time 23:23:23 url http://www.thawte.com`

Show the State of the Trustpool Policy

Use the following command to see the current state of the trustpool policy:

```
show crypto ca trustpool policy
```

This command returns information like the following:

```
0 trustpool certificates installed
Trustpool auto renewal statistics:
 State: Not in progress
 Last import result: Not attempted N/A
 Current Jitter: 0

Trustpool auto import statistics:
 Last import result: N/A
 Next schedule import at 22:00:00 Tues Jul 21 2015

Trustpool Policy

 Trustpool revocation checking is disabled.
 CRL cache time: 60 seconds
 CRL next update field: required and enforced
 Auto import of trustpool is enabled
 Automatic import URL: http://www.cisco.com/security/pki/trs/ios_core.p7b
 Download time: 22:00:00

Policy Overrides:
 None configured
```

Clear CA Trustpool

To reset the trustpool policy to its default state, use the following command:

```
clear configure crypto ca trustpool
```

Since the automatic import of trustpoint certificates is turned off by default, using this command disables the feature.

Set a Certificate Expiration Alert (for Identity or CA Certificates)

ASA checks all the CA and ID certificates in the trust points for expiration once every 24 hours. If a certificate is nearing expiration, a syslog will be issued as an alert.

A CLI is provided to configure the reminder and recurrence intervals. By default, reminders start at 60 days prior to expiration and recur every 7 days. You can configure the interval at which reminders are sent and the number of days before the expiration at which the first alert is sent by using the following command:

```
[no] crypto ca alerts expiration [begin <days before expiration>] [repeat <days>]
```

Irrespective of the alerts configuration, a reminder is sent every day during the last week of expiration. The following **show** and **clear** commands have also been added:

```
clear conf crypto ca alerts
show run crypto ca alerts
```

In addition to the renewal reminder, if an already expired certificate is found in the configuration, a syslog is generated once every day to rectify the configuration by either renewing the certificate or removing the expired certificate.

For example, assume that the expiration alerts are configured to begin at 60 days and repeat every 6 days after that. If the ASA is rebooted at 40 days, an alert is sent on that day, and the next alert is sent on the 36th day.

Note Expiration checking is not done on trust pool certificates. The Local CA trust point is treated as a regular trustpoint for expiration checking too.

Monitoring Digital Certificates

See the following commands for monitoring digital certificate status:

- **show crypto ca server**

  This command shows local CA configuration and status.

- **show crypto ca server cert-db**

  This command shows user certificates issued by the local CA.

- **show crypto ca server certificate**

  This command shows local CA certificates on the console in base 64 format and the rollover certificate when available, including the rollover certificate thumb print for verification of the new certificate during import onto other devices.

- **show crypto ca server crl**

  This command shows CRLs.

- **show crypto ca server user-db**

This command 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**

  This command shows users who are eligible to enroll.

- **show crypto ca server user-db enrolled**

  This command shows enrolled users with valid certificates.

- **show crypto ca server user-db expired**

  This command shows users with expired certificates.

- **show crypto ca server user-db on-hold**

  This command shows users without certificates who are not allowed to enroll.

- **show crypto key** *name of key*

  This command shows key pairs that you have generated.

- **show running-config**

  This command shows local CA certificate map rules.

Examples

The following example shows an RSA general-purpose key:

```
ciscoasa/contexta(config)# show crypto key mypubkey rsa
Key pair was generated at: 16:39:47 central Feb 10 2010
Key name: <Default-RSA-Key>
 Usage: General Purpose Key
 Modulus Size (bits): 2048
 Storage: config
 Key Data:

  30820122 300d0609 2a864886 f70d0101 01050003 82010f00 3082010a 02820101
  00ea2c38 df9c606e ddb7b08a e8b0a1a8 65592d85 0711cac5 fceddee1 fa494297
  525fffc0 90da8a4c e696e44e 0646c661 48b3602a 960d7a3a 52dae14a 5f983603
  e1f33e40 a6ce04f5 9a812894 b0fe0403 f8d7e05e aea79603 2dcd56cc 01261b3e
  93bff98f df422fb1 2066bfa4 2ff5d2a4 36b3b1db edaebf16 973b2bd7 248e4dd2
  071a978c 6e81f073 0c4cd57b db6d9f40 69dc2149 e755fb0f 590f2da8 b620efe6
  da6e8fa5 411a841f e72bb8ea cf4bdb79 f4e57ff3 a940ce3b 4a2c7052 56c1d17b
  af8fe2e2 e58718c6 ed1da0f0 1c6f36eb 79eb1aeb f098b5c4 79e07658 a52d8c7a
  51ceabfb f8ade096 7217cf2d 3728077e 89441d89 9bf5f875 c8d2db39 c858bb7a
  7d020301 0001
```

The following example shows the local CA CRL:

```
ciscoasa(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:

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

History for Certificate Management

Table 1: History for Certificate Management

Feature Name	Platform Releases	Description
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.

Feature Name	Platform Releases	Description
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**}.
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**].

Feature Name	Platform Releases	Description
Reference Identities	9.6(2)	TLS client processing now supports rules for verification of a server identity defined in RFC 6125, Section 6. Identity verification will be done during PKI validation for TLS connections to the Syslog Server and the Smart Licensing server only. If the presented identity cannot be matched against the configured reference identity, the connection is not established. We added or modified the following commands: **crypto ca reference-identity**, **logging host**, **call home profile destination address**
Local CA Server	9.12(1)	To make the FQDN of the enrollment URL configurable instead of using the ASA's configured FQDN, a new CLI option is introduced. This new option is added to the smpt mode of crypto ca server. We deprecated Local CA Server and will be removing in a later release—When ASA is configured as local CA server, it is enabled to issue digital certificates, publish Certificate Revocation Lists (CRLs), and securely revoke issued certificates. This feature has become obsolete and hence the crypto ca server command is deprecated.
Local CA Server	9.13(1)	We removed the local ca server support. Thus, the **crypto ca server** command and its subcommands are removed. We removed the following commands: **crypto ca server** and all of its subcommands.
Modifications to the CRL Distribution Point commands	9.13(1)	The static CDP URL configuration commands are removed and moved to the match certificate command. New/Modified commands: **crypto-ca-trustpoint crl** and **crl url** were removed with other related logic. **match-certificate override-cdp** was introduced.
CRL cache size increased	9.13(1)	To prevent failure of large CRL downloads, the cache size was increased, and the limit on the number of entries in an individual CRL was removed. • Increased the total CRL cache size to 16 MB per context for multi-context mode. • Increased the total CRL cache size to 128 MB for single-context mode.

Feature Name	Platform Releases	Description
Restoration of bypass certificate validity checks option	9.15(1)	The option to bypass revocation checking due to connectivity problems with the CRL or OCSP server that was removed in 9.13(1) was restored. New/Modified commands: **revocation-check crl none**, **revocation-check ocsp none**, **revocation-check crl ocsp none**, and **revocation-check ocsp crl none** were restored.
Modifications to Match Certificate commands to support static CRL Distribution Point URL	9.15(1)	The static CDP URL configuration command allowed static CDPs to be mapped uniquely to each certificate in a chain that is being validated. However, only one such mapping was supported for each certificate. This modification allows statically configured CDPs to be mapped to a chain of certificates for authentication. New/Modified commands: **match certificate** *map* **override cdp** *seq* **url** *url* and **no match certificate** *map* **override cdp** *seq* **url** *url*
Modifications to the trustpoint keypair and crypto key generate commands	9.16(1)	Support for certificates with key sizes smaller than 2048 was removed. Any configuration using 512, 768 or 1024-bit options are transitioned to 2048 with due notification. Support to use SHA1 hashing algorithm for certification was removed. **Note** **crypto ca permit-weak-crypto** command was introduced to override these restrictions. The new key option - EDDSA was added to the existing RSA and ECDSA options.
Support for OCSP and CRL IPv6 URL	9.20(1)	Support to use IPv6 OCSP and CRL URLs were added. The IPv6 addresses must be enclosed in square brackets.