

Certificates

Digital certificates provide digital identification for authentication. Certificates are used for SSL (Secure Socket Layer), TLS (Transport Layer Security), and DTLS (Datagram TLS) connections, such as HTTPS and LDAPS. The following topics explain how to create and manage certificates.

- About Certificates, on page 1
- Configuring Certificates, on page 4

About 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. A digital certificate also includes a copy of the public key for the user or device. Certificates are used for SSL (Secure Socket Layer), TLS (Transport Layer Security), and DTLS (Datagram TLS) connections, such as HTTPS and LDAPS.

You can create the following types of certificate:

- Internal certificates—Internal identity certificates are certificates for specific systems or hosts. You can generate these yourself using the OpenSSL toolkit or get them from a Certificate Authority. You can also generate a self-signed certificate. When internal certificates expire or becomes invalid for any reason, you can regenerate it from the following CLISH CLI command:

```
> system support regenerate-security-keyring
  String  Certificate to be regenerated, default or fdm
```

- Internal Certificate Authority (CA) certificates—Internal CA certificates are certificates that the system can use to sign other certificates. These certificates differ from internal identity certificates with respect to the basic constraints extension and the CA flag, which are enabled for CA certificates but disabled for identity certificates. You can generate these yourself using the OpenSSL toolkit or get them from a Certificate Authority. You can also generate a self-signed internal CA certificate. If you configure self-signed internal CA certificates, the CA runs on the device itself.
- Trusted Certificate Authority (CA) certificates—A trusted 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.

Certificate Authorities (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. A CA can be a trusted third party, such as

VeriSign, or a private (in-house) CA that you establish within your organization. CAs are responsible for managing certificate requests and issuing digital certificates. For more information, see Public Key Cryptography, on page 2.

Public Key Cryptography

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.

You can learn more about digital certificates and public key cryptography through openssl.org, Wikipedia, or other sources. Having a firm understanding of SSL/TLS cryptography will help you establish secure connections to your device.

Certificate Types Used by Feature

You need to create the right type of certificate for each feature. The following features require certificates.

Identity Policies (Captive Portal)—Internal Certificate

(Optional.) Captive portal is used in identity policies. Users must accept this certificate when authenticating to the device for purposes of identifying themselves and getting their IP address associated with their usernames. If you do not supply a certificate, the device uses an automatically generated certificate.

Identity Realms (Identity Policies and Remote Access VPN)—Trusted CA Certificate

(Optional.) If you use an encrypted connection for your directory server, the certificate must be accepted to perform authentication with the directory server. Users must authenticate when prompted by identity and remote access VPN policies. A certificate is not needed if you do not use encryption for the directory server.

Management Web Server (Management Access System Settings)—Internal Certificate

(Optional.) Firewall Device Manager is a web-based application, so it runs on a web server. You can upload a certificate that your browser accepts as valid to avoid getting an Untrusted Authority warning.

Remote Access VPN—Internal Certificate

(Required.) The internal certificate is for the outside interface, which establishes the device identity for Secure Clients when they make a connection to the device. Clients must accept this certificate.

Site-to-Site VPN—Internal and Trusted CA Certificates

If you use certificate authentication for a site-to-site VPN connection, you need to select the internal identity certificate used to authenticate the local peer in the connection. Although it is not part of the

VPN connection definition, you also need to upload the trusted CA certificates that were used to sign the local and remote peer identity certificates, so that the system can authenticate the peers.

SSL Decryption Policy—Internal, Internal Certificate, and Trusted CA Certificates and Certificate Groups

(Required.) The SSL decryption policy uses certificates for the following purposes:

- Internal certificates are used for known key decryption rules.
- Internal CA certificates are used for decrypt re-sign rules when creating the session between the client and Firewall Threat Defense device.
- Trusted CA certificates are used indirectly for decrypt re-sign rules when creating the session between the Firewall Threat Defense device and server. Trusted CA certificates are used to verify the signing authority of the server's certificate. You can configure these certificates directly or in a certificate group in the policy settings. The system includes a large number of trusted CA certificates, which are collected in the Cisco-Trusted-Authorities group, so you might not need to upload any additional certificates.

Streaming Telemetry—Internal Certificate and Trusted CA Certificate.

These certificates should be signed by the same CA to establish communication. The internal certificate represents the Firewall Threat Defense device. The trusted CA certificate represents the external telemetry collector.

Example: Generating an Internal Certificate using OpenSSL

The following example uses OpenSSL commands to generate an internal server certificate. You can obtain OpenSSL from openssl.org. Consult OpenSSL documentation for specific information. The commands used in this example might change, and you might have other options available that you might want to use.

This procedure is meant to give you an idea of how to obtain a certificate to upload to Firewall Device Manager.



Note The OpenSSL commands shown here are examples only. Adjust the parameters to fit your security requirements.

Procedure

Step 1 Generate a key.

```
openssl genrsa -out server.key 4096
```

Step 2 Generate a certificate signing request (CSR).

```
openssl req -new -key server.key -out server.csr
```

Step 3 Generate a self-signed certificate with the key and CSR.

```
openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

Because the Firewall Device Manager does not support encrypted keys, try to skip the challenge password by just pressing return when generating a self signed certificate.

Step 4 Upload the files into the appropriate fields when creating an internal certificate object in the Firewall Device Manager.

You can also copy/paste the file contents. The sample commands create the following files:

- server.crt—Upload or paste the contents into the Server Certificate field.
- server.key—Upload or paste the contents into the Certificate Key field. If you provided a password when generating the key, you can decrypt it using the following command. The output is sent to stdout, where you can copy it.

```
openssl rsa -in server.key -check
```

Configuring Certificates

Firewall Threat Defense supports X509 certificates in PEM or DER format. Use OpenSSL to generate certificates if needed, obtain them from a trusted Certificate Authority, or create self-signed certificates.

For more information on certificates, see About Certificates, on page 1.

For information on which type is used for each feature, see Certificate Types Used by Feature, on page 2.

The following procedure explains how you can create and edit objects directly through the Objects page. You can also create certificate objects while editing a certificate property by clicking the **Create New Certificate** link shown in the object list.

Procedure

Step 1 Select **Objects**, then select **Certificates** from the table of contents.

The system comes with the following pre-defined certificates, which you can use as is or replace.

- DefaultInternalCertificate
- DefaultWebserverCertificate
- NGFW-Default-InternalCA

The system also includes many trusted CA certificates from third party Certificate Authorities. These are used by SSL decryption policies for Decrypt Re-Sign actions. The Cisco-Trusted-Authorities group includes all of these certificates, and is the default group used by the SSL decryption policy.

You can click the pre-defined search filters to limit the list to just **System-defined** or **User-defined** certificates.

You can also use the **Weak Key** filter to find certificates whose keys are different from supported key lengths. The keys are considered weak even if they are longer than the supported lengths. We recommend that you replace these certificates with ones that have supported key lengths. The following are the supported lengths:

- RSA keys can be 2048, 3072, or 4096 bits.
- ECDSA keys can be 256, 384, or 521 bits.
- EDDSA keys can be 256 bits.

Step 2 Do one of the following:

- To create a new certificate object, use the command for the type of certificate from the + menu.
- To create a new certificate group, click  and select **Add Certificate Group**.
- To view or edit a certificate or group, click either the edit icon () or the view icon () for the certificate.
- To delete an unreferenced certificate or group, click the trash can icon () for the certificate.

For detailed information on creating or editing certificates, see the following topics:

- Uploading Internal and Internal CA Certificates, on page 5
- Generating Self-Signed Internal and Internal CA Certificates, on page 7
- Uploading Trusted CA Certificates, on page 8
- Configuring Trusted CA Certificate Groups, on page 10

Uploading Internal and Internal CA Certificates

Internal identity certificates are certificates for specific systems or hosts.

Internal CA certificates are certificates that the system can use to sign other certificates. These certificates differ from internal identity certificates with respect to the basic constraints extension and the CA flag, which are enabled for CA certificates but disabled for identity certificates.

You can generate these certificates yourself using the OpenSSL toolkit or get them from a Certificate Authority, and then upload them using the following procedure. For an example of generating a key, see Example: Generating an Internal Certificate using OpenSSL, on page 3.

You can also generate a self-signed internal identity and internal CA certificates. If you configure self-signed internal CA certificates, the CA runs on the device itself. For information on creating self-signed certificates, see Generating Self-Signed Internal and Internal CA Certificates, on page 7.

For information on the features that use these certificates, see Certificate Types Used by Feature, on page 2.

Procedure

Step 1 Select **Objects**, then select **Certificates** from the table of contents.

Step 2 Do one of the following:

- Click **+** > **Add Internal Certificate**, then click **Upload Certificate and Key**.
- Click **+** > **Add Internal CA Certificate**, then click **Upload Certificate and Key**.
- To edit or view a certificate, click the information icon ( ). The dialog box shows the certificate subject, issuer, and valid time range. Click Replace Certificate to upload a new certificate and key. You can also paste the certificate and key in the dialog box.

Step 3 Enter a **Name** for the certificate.

The name is used in the configuration as an object name only, it does not become part of the certificate itself.

Step 4 Click **Upload Certificate** (or **Replace Certificate** when editing) and select the certificate file (for example, *.crt). Allowed file extensions are .pem, .cert, .cer, .crt, and .der. Alternatively, paste in the certificate.

The certificate must be an X509 certificate in PEM or DER format.

The certificate you paste must include the BEGIN CERTIFICATE and END CERTIFICATE lines. For example:

```
-----BEGIN CERTIFICATE-----
MIICMTCCAZoCCQDdUV3NGK/cUjANBgkqhkiG9w0BAQsFADBdMQswCQYDVQQGEwJV
UzETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50ZXJuZXQgV2lkZ2l0
(...5 lines removed...)
shGJDReRYJQqilhHZrYTWZAYTrD7NQPHutK+ZiJng67cPgnNDuXEn55UwMOQoHBp
HMUwmhiGZlzJM8BpX2Js2yQ3ms30pr8rO+gPCPMCAwEAATANBgkqhkiG9w0BAQsF
AAOBgQCB02CebA6YjJCGr2CJZrQSeUwSveRBpmOuoqm98o2Z+5gJM5CkqgfxwCUn
RV7LRfQGFYd76V/5uor4Wx2ZCjqy6+zuQEm4ZxWNSZpA9UBixFXJCs9MBO4qkG5D
vlk3WYJfcgyJ10h4E4b0W2xiixBU+xoOTLRATnbKY36EWAG5cw==
-----END CERTIFICATE-----
```

Step 5 Click **Upload Key** (or **Replace Key** when editing) and select the certificate file (for example, *.key). The file extension must be .key. Alternatively, paste in the key for the certificate.

The key cannot be encrypted and it must be an RSA key.

For example:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQClSu1BknrMjzw/5FZ9YgdMLDUGJlbYgkjN7mVrkjyLQx2TYsem
r8iTiKB6iyTKbuS4iPeyEYkNF5FglCqKWEdmthNZkBhOsPs1A8e60r5mImeDrtw+
Cc0O5cSfnlTAw5CgcGkcxTCaGIZmXMkzwGlfYmzbJDeazfSmvys76A8I8wIDAQAB
AoGAUVDgEX8vXE0m9cOubPZ54pZo64KW/OJzUKP0TwxdLqGw/h39XFpkEXiIgmDL
(...5 lines removed...)
DSWvzekRDH83dmP66+MIbWePhbhty+D1OxbiuVuHV0/ZhxOhCG8tig3R8QJBAJmj
fId05+1dNI4tGbWv6hHh/H/dTP2STlZ3jERMZd29fjIRuJ9jpFC2lIDjvs8YGeAe
0YHkfSOULJn8/jOCf6kCQQDIJiHfGF/31Dk/8/5MGrg+3zau6oKXiuv6db8Rh+7l
MUOx09tvbBUy9REJq1YJWTKpeKD+E0QL+FX0bqvz4tHA
-----END RSA PRIVATE KEY-----
```

Step 6 Click **OK**.

If the key size is smaller than the minimum size we allow for generated self-signed certificates, you are warned that the certificate does not meet the recommended minimum requirements. Click **Proceed** to upload the certificate anyway, but we recommend that you create a new, stronger certificate.

Generating Self-Signed Internal and Internal CA Certificates

Internal identity certificates are certificates for specific systems or hosts.

Internal CA certificates are certificates that the system can use to sign other certificates. These certificates differ from internal identity certificates with respect to the basic constraints extension and the CA flag, which are enabled for CA certificates but disabled for identity certificates.

You can generate a self-signed internal identity and internal CA certificates, that is, the certificates are signed by the device itself. If you configure self-signed internal CA certificates, the CA runs on the device. The system generates both the certificate and the key.

You can also create these certificates using OpenSSL, or obtain them from a trusted CA, and upload them. For more information, see Uploading Internal and Internal CA Certificates, on page 5.

For information on the features that use these certificates, see Certificate Types Used by Feature, on page 2.

Procedure

Step 1 Select **Objects**, then select **Certificates** from the table of contents.

Step 2 Do one of the following:

- Click **+** > **Add Internal Certificate**, then click **Self-Signed Certificate**.
- Click **+** > **Add Internal CA Certificate**, then click **Self-Signed Certificate**.

Note

To edit or view a certificate, click the information icon (). The dialog box shows the certificate subject, issuer, and valid time range. Click **Replace Certificate** to upload a new certificate and key. When replacing a certificate, you cannot redo the self-signed characteristics explained in the following steps. Instead, you must paste or upload a new certificate as described in Uploading Internal and Internal CA Certificates, on page 5. The remaining steps apply to new self-signed certificates only.

Step 3 Enter a **Name** for the certificate.

The name is used in the configuration as an object name only, it does not become part of the certificate itself.

Step 4 Configure at least one of the following for the certificate subject and issuer information.

- **Country** (C)—The two-character ISO 3166 country code to include in the certificate. For example, the country code for the United States is US. Select the country code from the drop-down list.
- **State or Province** (ST)—The state or province to include in the certificate.
- **Locality or City** (L)—The locality to include in the certificate, such as the name of the city.
- **Organization** (O)—The organization or company name to include in the certificate.

- **Organizational Unit (Department)** (OU)—The name of the organization unit (for example, a department name) to include in the certificate.
- **Common Name** (CN)—The X.500 common name to include in the certificate. This could be the name of the device, web site, or another text string. This element is usually required for successful connections. For example, you must include a CN in the internal certificate used for remote access VPN.
- **Key Type**—The type of key to generate for this certificate: RSA, ECDSA (Elliptic Curve Digital Signature Algorithm), or EDDSA (Edwards-curve Digital Signature Algorithm).
- **Key Size**—The size of the key to generate. In general, longer keys are more secure. However, keys with larger modulus sizes take longer to generate and longer to process when exchanged. The allowed sizes differ based on the key type.
  - RSA keys can be 2048, 3072, or 4096 bits.
  - ECDSA keys can be 256, 384, or 521 bits.

    EDDSA keys can be 256 bits.
- **Validity Period**—How long the certificate will be considered valid. The default is 825 days from today, regardless of how you set the expiration date. Click **Set default** to return to the default. You can configure the period using one of the following methods. Be sure to replace certificates before they expire.
  - **By Date**—Click the **Expiration Date** and select the last day the certificate should be considered valid.
  - **By Number of Days**—Enter the number of days, starting today, that the certificate should be considered valid. After entering the number, you can click **By Date** to see the calculated expiration date.

Step 5 Click **Save**.

Uploading Trusted CA Certificates

A trusted Certificate Authority (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.

For information on the features that use these certificates, see Certificate Types Used by Feature, on page 2.

Obtain a trusted CA certificate from an external Certificate Authority, or create one using your own internal CA, for example, with OpenSSL tools. Then, use the following procedure to upload the certificate.

Before you begin

The system contacts Cisco once a day to determine if there are new or updated trusted CA certificates, and downloads updated certificates when available. This daily job ensures the pre-installed certificates are up to date. You can monitor this automated checking in the CLI using the **show cert-update** command. You can disable the daily job using the **configure cert-update auto-update disable** command, and manually download updates using the **configure cert-update run-now** command.

Procedure

Step 1 Select **Objects**, then select **Certificates** from the table of contents.

Step 2 Do one of the following:

- Click + > **Add Trusted CA Certificate**.
- To edit a certificate, click the edit icon ( ) for the certificate.

Step 3 Enter a **Name** for the certificate.

The name is used in the configuration as an object name only, it does not become part of the certificate itself.

Step 4 Click **Upload Certificate** (or **Replace Certificate** when editing) and select the trusted CA certificate file (for example *.pem). Allowed file extensions are .pem, .cert, .cer, .crt, and .der. Alternatively, paste in the trusted CA certificate.

The name of the server in the certificate must match the server Hostname / IP Address. For example, if you use 10.10.10.250 as the IP address but ad.example.com in the certificate, the connection fails.

The certificate must be an X509 certificate in PEM or DER format.

The certificate you paste must include the BEGIN CERTIFICATE and END CERTIFICATE lines. For example:

```
-----BEGIN CERTIFICATE-----
MIIFgTCCA2mgAwIBAgIJANvdcLnabFGYMA0GCSqGSIb3DQEBCwUAMFcxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJUWDEPMA0GA1UEBwwGYXVzdGluMRQwEgYDVQQKDAsx
OTIuMTY4LjEuMTEUMBIGA1UEAwwLMTkyLjE2OC4xLjEwHhcNMTYxMDI3MjIzNDE3
WhcNMTcxMDI3MjIzNDE3WjBXMQswCQYDVQQGEwJVUzELMAkGA1UECAwCVFgxDzAN
BgNVBAcMBmF1c3RpbjEUMBIGA1UECgwLMTkyLjE2OC4xLjExFDASBgNVBAMMCzE5
Mi4xNjguMS4xMIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEA5NceYwtP
ES6Ve+S9z7WLKGX5JlF58AvH82GPkOQdrixn3FZeWLQapTpJZt/vgtAI2FZIK31h
(...20 lines removed...)
hbr6HOgKlOwXbRvOdksTzTEzVUqbgxt5Lwupg3b2ebQhWJz4BZvMsZX9etveEXDh
PY184V3yeSeYjbSCF5rP71fObG9Iu6+u4EfHp/NQv9s9dN5PMffXKieqpuN20Ojv
2b1sfOydf4GMUKLBUMkhQnip6+3W
-----END CERTIFICATE-----
```

Step 5 Select **Skip CA Certificate Check** if this certificate was not issued by a Certificate Authority.

Skip the check if you need to install a local CA certificate as the trusted CA certificate.

Step 6 Set the **Validation Usage** to limit the use of the certificate.

Some features let you select whether a connection can be validated against a specific certificate. You must indicate in the certificate that these features can validly use the certificate, or a connection is refused.

Any feature not included in these options can validate against this certificate without an explicit usage allowance. For example, SSL decryption policies, and the web server that hosts the Firewall Device Manager, ignore the Validation Usage option. If you select any option in this field, the certificate will be downloaded to the running configuration that is displayed using the **show running-config** command.

The primary purpose of these options is to let you prevent VPN connections from getting established because they can be validated against a particular certificate.

- **SSL Server**—Validate the certificate on the remote SSL server. Used for dynamic DNS.
- **SSL Client**—Validate the certificate of the incoming remote access VPN connection.
- **IPsec Client**—Validate the certificate of the incoming IPsec site-to-site VPN connection.
- **No Validation Usage**—Validate features, such as LDAPS, that are not managed by the Snort inspection engine. Select this option only if you are having problems with a particular feature. This option is mutually exclusive with all other options: you must deselect it before you can select any other options, and deselect all options before you can select this option.

Step 7 Click **OK**.

Configuring Trusted CA Certificate Groups

Use external trusted CA certificate groups in the SSL decryption policy settings to specify which certificates the SSL decryption policy should trust. If an end user attempts a connection to a site whose certificate's issuer's certificate is not among the trusted certificates, the user gets a message asking to trust the certificate. So, not having the certificate in the trusted list is an inconvenience to the end user, but does not itself prevent the connection (which you might accomplish with access control rules).

The default group is Cisco-Trusted-Authorities. You would need to create your own group only if:

- You want to trust certificates that are not in the default group. You would then select both the default group and your new group in the SSL decryption policy settings.
- You want to trust a more limited list of certificates than the default group. You would then create a group that has a complete list of trusted certificates, not just your delta, and select it as the sole group in the SSL decryption policy settings.

Before you begin

Upload all of the trusted CA certificates that you will add to the group if they are not already in the system.

Procedure

Step 1 Select **Objects**, then select **Certificates** from the table of contents.

Step 2 Do one of the following:

- To create a new certificate group, click  and select **Add Certificate Group**.
- To edit a certificate group, click the edit icon () for the group.

Step 3 Enter a **Name** for the certificate group and optionally, a description.

Step 4 Click **+** to add certificates to the group.

Add all the certificates you need in the group. You can click **Create New Trusted CA Certificate** to upload new ones while you build the group.

If you no longer need a certificate in the group, click the X icon (on the right) for the certificate.

Step 5 Click **OK**.