Testing SSL Proxy Services

You can test or troubleshoot SSL proxy services by doing one of the following:

Generating a Self-Signed Certificate

You can generate multiple self-signed certificates for testing SSL proxy services, specifying the key label and the subject name, by entering the **test crypto pki self** command. You need to generate a key pair with a label before you generate the self-signed certificate. See the "Generating RSA Key Pairs" section on page 3-23 for details on generating key pairs.

After you enter the **test crypto pki self** command, you are prompted for the key pair label and the subject name of the certificate. A trustpoint with the key pair label as the trustpoint name is automatically created, and the hexadecimal dump of the self-signed certificate is displayed on the console. You can then assign the trustpoint to a proxy service for testing. You can repeat the procedure after reboot if necessary. The certificate is stored only in memory and cannot be saved in NVRAM as part of the configuration.

Note You cannot save the self-signed certificates as part of the configuration.

Note The **show crypto ca certificate** command does not display the self-signed certificates.

To generate a self-signed certificate and assign a trustpoint to the proxy service, perform this task:

	Command	Purpose
Step 1	`ssl-proxy# test crypto pki self`	Generates a self-signed certificate. **Note** After you enter this command, you are prompted for the subject name (using the LDAP format) and key pair name.
Step 2	`ssl-proxy# show crypto ca trustpoint label`	Displays information for the trustpoint.
Step 3	`ssl-proxy# configure terminal`	Enters configuration mode, selecting the terminal option.

	Command	Purpose
Step 4	`ssl-proxy(config)# ssl-proxy service` *proxy_name*	Defines the name of the SSL proxy service. **Note** The *proxy-name* is case-sensitive.
Step 5	`ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint` *trustpoint_label*	Assigns a trustpoint to a proxy service.
Step 6	`ssl-proxy(config-ssl-proxy)# end`	Exits configuration mode.
Step 7	`ssl-proxy# show ssl-proxy service` *proxy-name*	Displays the key pair and the serial number of the certificate chain used for a specified proxy service. **Note** The *proxy-name* is case-sensitive.

Note If the trustpoint already exists, it might be replaced by the test certificate. We recommend that you generate a unique key pair for the test certificate.

This example shows how to generate a key pair, generate a self-signed certificate, and assign the certificate to a proxy service:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# crypto key generate rsa general-keys label k1 modulus 1024
The name for the keys will be:k1

% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys ...[OK]

ssl-proxy(config)# end
ssl-proxy#
*Mar 20 14:34:01.543:%SYS-5-CONFIG_I:Configured from console by console
ssl-proxy#
ssl-proxy# test crypto pki self
Enter subject name for certificate
CN=testhost.my.com, O=lab, OU=testgroup
Enter name of key to be used
k1
ssl-proxy#
     30 82 02 06 30 82 01 6F 02 20 45 32 30 38 39 32
     45 37 38 31 42 41 46 45 45 45 44 45 37 37 41 36
     43 41 44 37 44 43 45 38 34 37 30 0D 06 09 2A 86
     48 86 F7 0D 01 01 04 05 00 30 3C 31 12 30 10 06
     03 55 04 0B 13 09 74 65 73 74 67 72 6F 75 70 31
     0C 30 0A 06 03 55 04 0A 13 03 6C 61 62 31 18 30
     16 06 03 55 04 03 13 0F 74 65 73 74 68 6F 73 74
     2E 6D 79 2E 63 6F 6D 30 1E 17 0D 30 33 30 33 32
     30 31 34 33 35 30 30 5A 17 0D 31 33 30 33 31 37
     31 34 33 35 30 30 5A 30 3C 31 12 30 10 06 03 55
     04 0B 13 09 74 65 73 74 67 72 6F 75 70 31 0C 30
     0A 06 03 55 04 0A 13 03 6C 61 62 31 18 30 16 06
     03 55 04 03 13 0F 74 65 73 74 68 6F 73 74 2E 6D
     79 2E 63 6F 6D 30 81 9F 30 0D 06 09 2A 86 48 86
     F7 0D 01 01 01 05 00 03 81 8D 00 30 81 89 02 81
```

```
      81 00 EC 21 35 B5 0E BF 9C 1C 71 05 05 B2 8A 47
      C8 F9 13 6C 5A 14 77 63 BD 0C B7 D3 35 6A DB B8
      0F C2 D2 39 A8 62 67 EE CB BC 8D 5E F8 C2 1E 8E
      D6 39 62 07 B4 64 20 D8 29 25 1E 9E 06 C8 F8 F9
      A6 29 05 19 CC D9 00 E9 2D 96 6D CE CA E0 D7 BF
      DC 9D 1B 7E 71 C1 D7 3F 25 28 41 5A F9 FB 98 66
      B9 A7 81 18 79 71 2A AC 55 F8 CC A4 4A 90 35 A7
      E9 BD 79 66 BC 5B C5 98 16 B0 63 5B D3 6E 85 65
      42 1B 02 03 01 00 01 30 0D 06 09 2A 86 48 86 F7
      0D 01 01 04 05 00 03 81 81 00 A3 93 7A E6 60 54
      8C 3A FF 6A 72 A8 1F 4B AD 79 53 C4 37 DF C4 D4
      F9 F4 58 3C E4 D8 BE FF BB C5 F9 CD B0 20 7F 3D
      0E B5 11 8E FA 33 02 9E 5E 52 36 4D 0F AB 21 41
      97 A4 2D 94 4D DF D2 A0 B4 DE B0 2E 1C BA 16 A9
      4C 28 34 72 8E D5 82 F6 B6 B2 D6 4E B5 1A F0 BB
      6B 65 E7 85 52 72 9F 9C BC A7 D9 B4 79 AB 6B C2
      DC FD AD 02 D3 28 87 CD 06 8B 11 3C 22 85 28 1B
      DC 04 05 8D 4F 1D 07 8D D0 BC

ssl-proxy# show crypto ca trustpoint k1
Trustpoint k1:
    Subject Name:
    CN = testhost.my.com
     O = lab
     OU = testgroup
          Serial Number:453230383932453738314241464545454445373741364341443744434538343 7
    Application generated trust point

ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy service ser1
ssl-proxy(config-ssl-proxy)# certificate rsa general trustpoint k1
ssl-proxy(config-ssl-proxy)#
*Mar 20 14:36:09.567:%STE-6-PKI_SERVER_CERT_INSTALL:Proxy:ser1, Trustpoint:k1, Key:k1,
Serial#:453230383932453738314241464545454445373741364341443744434538343 7, Index:3
ssl-proxy(config-ssl-proxy)# end
ssl-proxy#
*Mar 20 14:36:16.363:%SYS-5-CONFIG_I:Configured from console by console
ssl-proxy#
ssl-proxy# show ssl-proxy service ser1
Service id:2, bound_service_id:258
Virtual IP address not configured
Server IP address not configured
rsa-general-purpose certificate trustpoint:k1
  Certificate chain for new connections:
    Server Certificate:
       Key Label:k1
       Serial Number:453230383932453738314241464545454445373741364341443744434538343 7
       Self-signed
Certificate chain complete
Admin Status:down
Operation Status:down
ssl-proxy#
```

Importing the Embedded Test Certificate

A test PKCS12 file (test/testssl.p12) is embedded in the SSL software on the module. You can install the file into NVRAM for testing purposes and for proof of concept. After the PKCS12 file is installed, you can import it to a trustpoint, and then assign it to a proxy service that is configured for testing.

To install and import the test file, perform this task:

	Command	Purpose
Step 1	`ssl-proxy# test ssl-proxy certificate install`	Installs the test PKCS12 file to NVRAM.
Step 2	`ssl-proxy(config)# crypto ca import` *trustpoint_label* `pkcs12 nvram:test/testssl.p12` *passphrase*	Imports the test PKCS12 file to the module. **Note** For the test certificate, the *passphrase* is **sky is blue**.
Step 3	`ssl-proxy(config)# ssl-proxy service` *test_service*	Defines the name of the test proxy service.
Step 4	`ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint` *trustpoint_label*	Applies a trustpoint configuration to the proxy server.
Step 5	`ssl-proxy# show ssl-proxy stats` *test_service*	Displays test statistics information.

This example shows how to import the test PKCS12 file:

```
ssl-proxy# test ssl-proxy certificate install
% Opening file, please wait ...
% Writing, please wait ............
% Please use the following config command to import the file.
  "crypto ca import <trustpoint-name> pkcs12 nvram:test/testssl.p12 sky is blue"
% Then you can assign the trustpoint to a proxy service for testing.

ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# crypto ca import test-tp pkcs12 nvram:test/testssl.p12 sky is blue
Source filename [test/testssl.p12]?
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy service test-service
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint test-tp
ssl-proxy(config-ssl-proxy)# end
ssl-proxy#
```