



Configuring Back-End SSL

This chapter describes the steps required to configure back-end SSL on a CSS. It contains the following major sections:

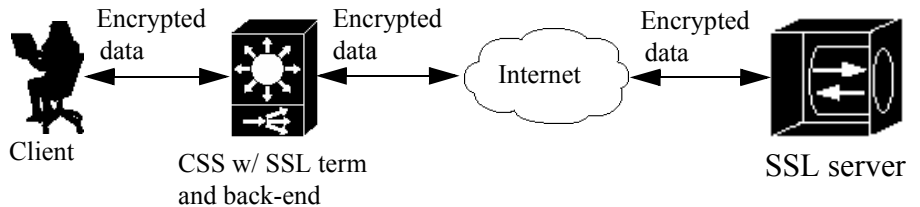
- [Overview of Back-End SSL](#)
- [Creating an SSL Proxy List](#)
- [Adding a Description to an SSL Proxy List](#)
- [Configuring Back-End SSL Servers in an SSL Proxy List](#)
- [Activating and Suspending an SSL Proxy List](#)
- [Modifying an SSL Proxy List](#)
- [Configuring a Service for Back-End SSL](#)
- [Configuring a Content Rule for Back-End SSL](#)

Overview of Back-End SSL

Back-end SSL allows a CSS to initiate a connection with an SSL server. When used with SSL termination, back-end SSL provides a secure end-to-end connection between a client and an SSL server.

Figure 5-1 illustrates back-end SSL with SSL termination.

Figure 5-1 Back-End SSL with SSL Termination



An SSL proxy list determines the flow of SSL information among the SSL module, the client, and the server. An SSL proxy list comprises one or more back-end SSL servers (related by index entry). The back-end SSL server entry initiates the connection to an SSL server. You can define a maximum of 256 virtual or back-end SSL servers for a single SSL proxy list.

After you create and configure the entries in a proxy list, you must activate the list, and then add the SSL proxy list to a service to initiate the transfer of SSL configuration data to the SSL module. When you activate the service, the CSS transfers the data to the module. Then add each SSL service to an SSL content rule.

Creating an SSL Proxy List

An SSL proxy list is a group of related back-end SSL servers that are associated with an SSL service. To create an SSL proxy list, use the **ssl-proxy-list** command.

You can access the `ssl-proxy-list` configuration mode from most configuration modes except for `ACL`, `boot`, `group`, `rmon`, or `owner` configuration modes. You can also use this command from the `ssl-proxy-list` configuration mode to access another SSL proxy list. Enter the SSL proxy list name as an unquoted text string from 1 to 31 characters.

For example, to create the SSL proxy list, `ssl_list1`, enter:

```
(config)# ssl-proxy-list ssl_list1  
Create ssl-list <ssl_list1>, [y/n]: y
```

Once you create an SSL proxy list, the CLI enters into the `ssl-proxy-list` configuration mode.

```
(config-ssl-proxy-list[ssl_list1])#
```

To delete an existing SSL proxy list, enter:

```
(config)# no ssl-proxy-list ssl_list1  
Delete ssl-list <ssl_list1>, [y/n]: y
```



Note

You cannot delete an SSL proxy list if an SSL service is in use and contains the active SSL proxy list. You must first suspend the SSL service to delete a specific SSL proxy list.

Adding a Description to an SSL Proxy List

To specify a description for an SSL proxy list, use the **description** command. Enter the description as a quoted text string with a maximum of 64 characters, including spaces.

For example, to add a description to the `ssl_list1` SSL proxy list, enter:

```
(config-ssl-proxy-list[ssl_list1])# description "This is the SSL list  
for www.brandnewproducts.com"
```

To remove the description from a specific SSL proxy list, enter:

```
(config-ssl-proxy-list[ssl_list1])# no description
```

Configuring Back-End SSL Servers in an SSL Proxy List

This section discusses creating one or more back-end SSL servers for an SSL proxy list. Use the **backend-server** command to define an index entry in the SSL proxy list that you then use to configure specific SSL parameters associated with the SSL proxy list. An SSL module in the CSS uses the SSL proxy list to initiate a connection to a back-end SSL server. You must define a back-end server index number before configuring SSL proxy list parameters. You can define a maximum of 256 back-end SSL servers for a single SSL proxy list.



Note

No modifications to an SSL proxy list are permitted on an active list. Suspend the list prior to making changes, and then reactivate the SSL proxy list once the changes are complete. The CSS sends any additions or changes to any active SSL services using the proxy list. For more information, see the [“Modifying an SSL Proxy List”](#) section.

To configure a back-end server for use by the SSL module, you must create and configure a back-end server entry in an SSL proxy list. Configure an IP address that corresponds to the address of the service and the server IP address. Then activate the SSL proxy list.

After you configure and activate the SSL proxy list, add the list to a back-end SSL service; assign a service type of **ssl-accel-backend**. When you activate the service, the CSS sends the configuration data to the SSL module.

The following sections describe:

- [Creating a Back-End SSL Server in an SSL Proxy List](#)
- [Configuring the VIP Address for an SSL Back-End Server](#)
- [Configuring the Virtual Port](#)
- [Configuring the Server IP Address](#)
- [Configuring the Server Port](#)

- [Configuring SSL Version](#)
- [Configuring the Available Cipher Suites](#)
- [Configuring SSL Session Cache Timeout](#)
- [Configuring SSL Session Handshake Renegotiation](#)
- [Configuring TCP Virtual Client Connections Timeout Values](#)
- [Configuring TCP Server-Side Connection Timeout Values on the SSL Module](#)
- [Changing the Acknowledgement Delay for SSL TCP Connections](#)
- [Specifying the Nagle Algorithm for SSL TCP Connections](#)
- [Specifying the TCP buffering for SSL TCP Connections](#)

Creating a Back-End SSL Server in an SSL Proxy List

You must create a back-end SSL server before you can configure back-end SSL proxy-list parameters. To create a back-end server in the SSL proxy list, use the **backend-server** *number* command. This command assigns it a number (index entry) in the SSL proxy list that you use to configure specific SSL parameters associated with the back-end SSL server (for example, VIP address, certificate name, and key pair). Enter a value from 1 to 256.

For example, to create back-end server 1 in the proxy list, enter:

```
(config-ssl-proxy-list[ssl_list3])# backend-server 1
```

To remove back-end server 1 from the SSL proxy list, enter:

```
(config-ssl-proxy-list[ssl_list3])# no backend-server 1
```

Configuring a Back-End SSL Server Type

By default, a back-end SSL server has a type of **backend-ssl** that allows a CSS to:

- Receive encrypted data from a client
- Decrypt the data for load balancing
- Re-encrypt the data and send it to an SSL server over an SSL connection

If you configure an SSL initiation server but want to reconfigure it as a back-end SSL server in the same proxy list, use the **backend-server *number* type backend-ssl** command.

For example, to reconfigure SSL initiation server 1 as a back-end SSL server in SSL proxy list `ssl_list3`, enter:

```
(config-ssl-proxy-list[ssl_list3])# backend-server 1 type backend-ssl
```

For information about SSL initiation, see [Chapter 5, Configuring SSL Initiation](#).

Configuring the VIP Address for an SSL Back-End Server

To configure the VIP address for the back-end server, use the **backend-server *number* ip address** command. The VIP address corresponds to the address of the service.

For example, to configure the VIP address 192.168.2.3 for back-end server 1, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 ip address
192.168.2.3
```

To remove the VIP address from the back-end server, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 ip address
```



Note

If you have not configured the VIP address when you issue the **active** command, the following error message appears and the CSS does not activate the list.

```
SSL-server/Backend-server must have valid IP Address
```

Configuring the Virtual Port

By default, the virtual port for the back-end server is port 80. The virtual port directs the clear text data traffic from the SSL module to the CSS. To configure a different virtual port for the SSL back-end server, use the **backend-server number port** command. Enter a port number from 1 to 65535.

**Note**

If you configure the **backend-server number ip address** and **server-ip** commands with the same address, configure the **backend-server number port** and **server-port** commands with different port numbers.

For example, to configure a port number of 1200, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 port 1200
```

To reset the port to the default value of 80, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 port
```

Configuring the Server IP Address

The server IP address is the IP address for the back-end SSL server. To configure the server IP address for the back-end server, use the **backend-server number server-ip** command.

For example, to configure the server IP address 192.168.2.3, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 server-ip  
192.168.2.3
```

To remove the server IP address from the back-end server, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 server-ip
```

**Note**

If you do not configure the server IP address when you issue the **active** command, the following error message appears and the CSS does not activate the list.

```
SSL-server/Backend-server must have valid IP address
```

Configuring the Server Port

By default the server port for the back-end SSL server is port 443. To configure a different server port for the SSL back-end server, use the **backend-server number server-port** command. Enter a port number from 1 to 65535.



Note

If you configure the **backend-server number ip address** and **server-ip** commands with the same address, configure the **backend-server number port** and **server-port** commands with different port numbers.

For example, to configure the server port number 155, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 server-port 155
```

To reset the port to the default value of 443, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 server-port
```

Configuring SSL Version

For a back-end server, the SSL module initiates the SSL connection. The version in the ClientHello message sent to the server indicates the highest supported version.

By default, the SSL version is SSL version 3 and TLS version 1. The SSL module sends a ClientHello that has an SSL version 3 header with the ClientHello message set to TLS version 1.

Use the **backend-server number version** command to specify which version of SSL the back-end server supports:

- **ssl3** - SSL version 3.
- **tls1** - TLS version 1.
- **ssl-tls** - SSL version 3 and TLS version 1. The SSL module sends a ClientHello that has an SSL version 3 header with the ClientHello message set to TLS version 1.

For example, to configure the SSL version 3, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 version ssl3
```


To reset the default SSL version, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 version
```

Configuring the Available Cipher Suites

To configure one or more specific cipher suites to be used by the back-end server, use the **backend-server *number* cipher** command. By default, all supported hardware accelerated cipher suites are enabled.

[Table 4-1](#) earlier in this chapter lists all supported cipher suites for the SSL module and the corresponding cipher suite value. These values match those defined for SSL version 3.0 and TLS version 1.0. The table also lists those Cipher suites that are exportable in any version of the software.

If you use the default setting or select the **all-cipher-suite** option, the CSS sends the suites in the same order as they appear in [Table 4-1](#), starting with `rsa-with-rc4-128-md5`.



Note

The **all-cipher-suites** option reenables all cipher suites for the back-end server. This option works only when you do not configure specifically-defined ciphers. To return to using the **all-cipher-suites** option, you must remove all specifically-defined ciphers.

For example, to configure a cipher of `rsa-with-rc4-128-md5`, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 cipher  
rsa-with-rc4-128-md5
```

When negotiating which cipher suite to use, the SSL module sends the ciphers in weighted order to the server with the highest weighted cipher first in the list.

By default, all configured cipher suites have a weight of 1. Optionally, you can assign a priority weight to the cipher suite, with 10 being the highest.

For example, to set a weight of 10 to a cipher suite, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 cipher  
rsa-with-rc4-128-md5 weight 10
```

To remove one or more of the configured cipher suites for the back-end server, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 cipher
rsa-with-rc4-128-md5
```

Configuring SSL Session Cache Timeout

In SSL, every time a client and server go through a full key exchange and establish a new master secret key, a new session is created. Enabling a session cache timeout allows the reuse of the master key on subsequent connections by the client. When you disable the cache timeout, the full SSL handshake must occur on each new connection to the SSL module (the virtual client). Use the **backend-server number session-cache** command to configure the SSL module to resume connection with a back-end SSL server using a previously established secret key.

By default, the cache timeout is enabled with a timeout of 300 seconds (5 minutes). The timeout value can range from 0 to 72000 (0 seconds to 20 hours). A timeout value of 0 disables the session cache reuse.

For example, to configure the SSL session cache timeout of 500 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 session-cache 500
```

To reset the session cache ID reuse to the default of enabled with a timeout of 300 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 session-cache
```

To disable session cache ID reuse, enter a timeout value of 0 seconds:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 session-cache 0
```

Configuring SSL Session Handshake Renegotiation

The SSL session handshake commands send the SSL HelloRequest message to a client to restart SSL handshake negotiation. SSL rehandshake is useful when a connection has been established for a lengthy period of time and you want to ensure security by reestablishing the SSL session between the CSS and the back-end SSL server.

Use the **backend-server number handshake data kbytes** command to force an SSL rehandshake after the exchange of a certain amount of data between the CSS and the back-end SSL server, after which the CSS transmits the SSL handshake message and reestablishes the SSL session.

By default, the SSL rehandshake is disabled (set to 0) for a back-end SSL server after the exchange of data. The data value is in kilobytes and is from 0 to 512000 kilobytes.

For example, to configure the SSL session rehandshake data value of 500 Kbytes, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 handshake data 500
```

To reset the rehandshake data value to 0, disable the rehandshake after the exchange of data. For example, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 handshake data
```

Use the **backend-server number handshake timeout seconds** command to specify a maximum timeout value, after which the CSS transmits the SSL handshake message and reestablishes the SSL session. Setting a timeout value forces the SSL session to renegotiate a new session key after a session has lasted the defined number of seconds. The selection of an SSL rehandshake timeout value is important when using the **advanced-balance ssl** load-balancing method for a Layer 5 content rule to fine-tune the SSL session ID used to stick the client to the server.

By default, the SSL rehandshake timeout is disabled (set to 0) for the back-end SSL server. The timeout value is from 0 to 72000 (0 seconds to 20 hours).

For example, to configure a 30-second timeout of an SSL session rehandshake, enter:

```
(config-ssl-proxy-list[ssl_list1])# back-end-server 1 handshake timeout 30
```

To reset the timeout to 0, disable the rehandshake timeout period for the back-end server by entering:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 handshake timeout
```

Configuring TCP Virtual Client Connections Timeout Values

The TCP connection between the client and the SSL module is terminated when the specified time interval elapses. The TCP timeout functions enable you to have more control over the TCP connection between the client and the SSL module.

To configure the TCP connection with the client, see the following sections:

- [Specifying a TCP SYN Timeout Value for the Virtual Client Connection](#)
- [Specifying a TCP Inactivity Timeout for a Virtual Client Connection](#)

Specifying a TCP SYN Timeout Value for the Virtual Client Connection

The CSS SYN timer counts the delta between the CSS sending the SYN/ACK and the client replying with an ACK as the means to terminate the TCP three-way handshake. Use the **backend-server number tcp virtual syn-timeout seconds** command to specify a timeout value that the CSS uses to terminate a TCP connection with a client and the SSL module that has not successfully completed the TCP three-way handshake prior to transferring data.

Enter a TCP SYN inactivity timeout value in seconds, from 0 (TCP SYN timeout disabled) to 3600 (1 hour). The default is 30 seconds. When you set the command to 0, the timer becomes inactive and the retransmit timer eventually terminates a broken TCP connection.



Note

The connection timer should always be less than the retransmit termination time for new SSL and TCP connections.

To configure the TCP SYN timeout of 100 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp virtual
syn-timeout 100
```

To disable the timeout, set the value to 0:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp virtual
syn-timeout 0
```

To reset the timeout to the default value of 30 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 tcp virtual
syn-timeout
```

Specifying a TCP Inactivity Timeout for a Virtual Client Connection

The TCP inactivity timeout begins once the CSS receives an ACK from the client to terminate the TCP three-way handshake. The inactivity timer resumes immediately following where the SYN timer stops, with regard to traffic flow. Use the **backend-server *number* tcp virtual inactivity-timeout *seconds*** command to specify a timeout value that the CSS uses to terminate a TCP connection with the client and the SSL module when there is little or no activity occurring on the connection.

Enter a TCP inactivity timeout value in seconds, from 0 (TCP inactivity timeout disabled) to 3600 (1 hour). The default is 240 seconds.

Based on the default parameters for retransmission, the timer value should be larger than 60 seconds (1 minute).

For example, to configure the TCP inactivity timeout period of 100 seconds for the virtual client connection, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp virtual  
inactivity-timeout 100
```

To disable the timeout, set the value to 0:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp virtual  
inactivity-timeout 0
```

To reset the timeout to the default value of 240 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 tcp virtual  
inactivity-timeout
```

Configuring TCP Server-Side Connection Timeout Values on the SSL Module

The TCP connection between the SSL module and a server is terminated when the specified time interval elapses. The TCP timeout functions enable you to have more control over TCP connections between the CSS SSL module and a server.

To configure the timeout values of a TCP connection with the server, see the following sections:

- [Specifying a TCP SYN Timeout Value for a Server-Side Connection](#)
- [Specifying a TCP Inactivity Timeout for a Server-Side Connection](#)

Specifying a TCP SYN Timeout Value for a Server-Side Connection

The TCP SYN timer counts the delta between the CSS initiating the back-end TCP connection by transmitting a SYN and the server replying with a SYN/ACK. Use the **backend-server number tcp server syn-timeout seconds** command to specify a timeout value that the CSS uses to end a TCP connection with a server that has not successfully completed the TCP three-way handshake prior to transferring data.

Enter a TCP SYN timeout value in seconds, from 0 (TCP SYN timeout disabled) to 3600 (1 hour). The default is 30 seconds. When you set the command to 0, the timer becomes inactive and the retransmit timer eventually terminates a broken TCP connection.

**Note**

The connection timer should always be less than the retransmit termination time for new SSL and TCP connections.

For example, to configure the TCP SYN timeout of 100 seconds for the server-side connection, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp server  
syn-timeout 100
```

To disable the timeout, set the value to 0:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp server  
syn-timeout 0
```

To reset the timeout to the default value of 30 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 tcp server  
inactivity-timeout
```

Specifying a TCP Inactivity Timeout for a Server-Side Connection

The TCP inactivity timeout begins once the CSS receives a SYN/ACK from the server. The inactivity timer resumes immediately following where the SYN timer stops, with regard to traffic flow. Use the **backend-server *number* tcp server inactivity-timeout *seconds*** command to specify a timeout value that the CSS uses to terminate a TCP connection with a server when there is little or no activity occurring on the connection.

Enter a TCP inactivity timeout value in seconds, from 0 (TCP inactivity timeout disabled) to 3600 (1 hour). The default is 240 seconds.

For example, to configure the TCP inactivity timeout period of 100 seconds for the server-side connection, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp server  
inactivity-timeout 100
```

To disable the timeout, set the value to 0:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp server  
inactivity-timeout 0
```

To reset the timeout to the default value of 240 seconds, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 1 tcp server  
inactivity-timeout
```

Changing the Acknowledgement Delay for SSL TCP Connections

By default, the acknowledgement delay on a client or server connection is 200 milliseconds (ms). You can disable or adjust the SSL TCP timer length for delayed acknowledgements by using the following command:

```
backend-server server-num tcp virtualserver ack-delay value
```

The *value* variable is the timer length in milliseconds (ms) for delayed acknowledgements. The default value is 200. Enter a value from 0 to 10000. A value of 0 disables the acknowledgement delay in receiving SSL traffic from the client. Disabling the timer improves the performance for sessions using the SSL session cache (Session ID Reuse).

For example, to set an acknowledgement delay of 400 ms for the TCP connection between the client and the SSL module, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 20 tcp virtual  
ack-delay 400
```

To set an acknowledgement delay of 400 ms for the TCP connection between the server and the SSL module, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 20 tcp server  
ack-delay 400
```

Specifying the Nagle Algorithm for SSL TCP Connections

The TCP Nagle algorithm automatically concatenates a number of small buffer messages transmitted over the TCP connection between a client and the SSL module or between a back-end server and the SSL module. This process increases the throughput of your CSS by decreasing the number of packets sent over each TCP connection. However, the interaction between the Nagle algorithm and the TCP delay acknowledgment may increase latency in your TCP connection. Disable the Nagle algorithm when you observe an unacceptable delay in a TCP connection (clear-text or SSL).

Use the **backend-server *number* tcp virtual nagle** command to disable or reenble the Nagle algorithm for the TCP connection between the client and the SSL module. The syntax for this command is:

backend-server *number* tcp virtual nagle enable|disable

To disable the Nagle algorithm for the TCP connection between the client and the SSL module, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp virtual nagle disable
```

To reenble the Nagle algorithm for the TCP connection between the client and the SSL module, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp virtual nagle enable
```

Use the **backend-server *number* tcp server nagle** command to disable or reenble the Nagle algorithm for the TCP connection between the server and the SSL module. The syntax for this command is:

backend-server *number* tcp server nagle enable|disable

To disable the Nagle algorithm for the TCP connection between the server and the SSL module, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp server nagle disable
```

To reenble the Nagle algorithm for the TCP connection between the server and the SSL module, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 1 tcp server nagle enable
```

Specifying the TCP buffering for SSL TCP Connections

If the network is slow and congested, you can increase the buffer size that the CSS buffers for a given TCP connection before shutting down the TCP window to 0. Use the **backend-server** *number* **tcp buffer-share** command to set the TCP buffering from the client or server on a given connection. The syntax for this command is:

```
backend-server number tcp buffer-share rx number1 | tx number2
```

To set the amount of data in bytes that a given connection can buffer from the client traffic, use the **rx** *number1* keyword and variable. By default, the buffer size is 32768. The buffer size can range from 16400 to 262144. For example, to set the value to 65536, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 20 tcp buffer-share  
rx 65536
```

To reset the reset the buffer size to the default of 32768, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 20 tcp  
buffer-share rx
```

To set the amount of data in bytes that a given connection can buffer from the server to the client, use the **tx** *number2* keyword and variable. By default, the buffer size is 65536. The buffer size can range from 16400 to 262144. For example, to set the value to 131072, enter:

```
(config-ssl-proxy-list[ssl_list1])# backend-server 20 tcp buffer-share  
tx 131072
```

To reset the reset the buffer size to the default of 65536, enter:

```
(config-ssl-proxy-list[ssl_list1])# no backend-server 20 tcp
buffer-share tx
```

Activating and Suspending an SSL Proxy List

Before you can activate an SSL proxy list, ensure that you have created at least one virtual or back-end SSL server in the list (see the [“Configuring Virtual SSL Servers for an SSL Proxy List”](#) section or the [“Specifying the Nagle Algorithm for SSL TCP Connections”](#) section earlier in this chapter).

The CSS checks the SSL proxy list to verify that all of the necessary components are configured, including verification of the certificate and key pair against each other. If the verification fails, the certificate name is not accepted and the CSS logs the error message `Certificate and key pair do not match` and does not activate the SSL proxy list. You must either remove the configured key pair or configure an appropriate certificate.

Use the **active** command to activate the new or modified SSL proxy list. For example, enter:

```
(config-ssl-proxy-list[ssl_list1])# active
```

After you activate an SSL proxy list, you can add it to a service. See the [“Configuring a Service for Back-End SSL”](#) section later in this chapter.

To view the virtual or back-end SSL servers in a list, use the **show ssl-proxy-list** (see [Chapter 7, Displaying SSL Configuration Information and Statistics](#)).

Use the **suspend** command to suspend an active SSL proxy list.

To suspend an active SSL proxy list, enter:

```
(config-ssl-proxy-list[ssl_list1])# suspend
```

Modifying an SSL Proxy List

You cannot modify an SSL proxy list when the list is active. Suspend the list prior to making changes, and then reactivate the SSL proxy list once you complete the changes.

After you modify the proxy list, you do not need to suspend and reactivate the SSL services using the list. The SSL module:

- Sends any additions or changes to any active SSL services using the proxy list.
- Clears the connections for the SSL-related services that were modified or deleted. If the SSL module receives any packets for those connections, the module sends a TCP RST.



Caution

If you are using front-end and back-end servers for a flow, both servers must be active for the end-to-end connection to work. If you modify an SSL proxy list, do not delete the back-end server from the list when the service is still active. The end-to-end connection will fail when you reactivate the SSL proxy list.

Configuring a Service for Back-End SSL

An SSL proxy list may belong to multiple SSL services (one SSL proxy list per service), and an SSL service may belong to multiple content rules. You can apply the services to content rules that allow the CSS to direct SSL requests for content.



Note

The CSS supports one active SSL service for each SSL module in the CSS, one SSL service per slot. You can configure more than one SSL service for a slot but only a single SSL service can be active at a time.

The requirements for the type of service to be added to the back-end content rule is as follows:

- The service must have a configured IP address
- The keepalive type for a back-end service can be none, ICMP, TCP, SSL, named, scripted, or encrypted HTTP. Encrypted HTTP keepalives can be persistent or non-persistent.

If you configure a TCP, SSL, or encrypted keepalive, you must configure the keepalive port correctly for the service to work.

- You must configure an SSL proxy list that contains back-end-server configuration for this type of service.

**Note**

If you do not configure a service port, the CSS uses the same port number as the back-end content rule.

This section covers:

- [Creating an SSL Service](#)
- [Configuring the Back-End SSL Service Type](#)
- [Adding an SSL Proxy List for a Back-End SSL Server](#)
- [Configuring the Back-End Service Keepalive Type](#)
- [Configuring an IP Address for a Back-End SSL Service](#)
- [Configuring the Port Number for a Back-End SSL Service](#)
- [Activating the SSL Service](#)
- [Suspending the SSL Service](#)

Creating an SSL Service

When creating a service for use with an SSL module, you must identify it as an SSL service for the CSS to recognize it. For additional details on creating a service, refer to the *Cisco Content Services Switch Content Load-Balancing Configuration Guide*.

Enter the SSL service name, from 1 to 31 characters.

To create service *ssl_serv1*, enter:

```
(config)# service ssl_serv1  
Create service <ssl_serv1>, [y/n]: y
```

The CSS transitions into the newly created service mode.

```
(config-service[ssl_serv1])#
```

Configuring the Back-End SSL Service Type

You must configure the **ssl-accel-backend** service type for a back-end SSL service. To configure a service type for a back-end SSL service, enter:

```
(config-service[server1])# type ssl-accel-backend
```

Adding an SSL Proxy List for a Back-End SSL Server

After you configure an SSL proxy list for a back-end SSL server, add the active list to an SSL service to define how the CSS processes SSL requests for content from a back-end SSL server. Configuring the back-end SSL service is similar to configuring a local service except you must set the service type to **ssl-accel-backend**. Also, this type of service requires an SSL proxy list with a back-end server entry.

An SSL proxy list contains the parameters for the back-end SSL service. To add the proxy list to the service, use the **add ssl-proxy-list** command. For more information on configuring an SSL proxy list for a back-end server, see the [“Configuring the Back-End SSL Service Type”](#) section earlier in this chapter.

Enter the name of the previously created SSL proxy list (see the [“Creating an SSL Proxy List”](#) section in this chapter) that you want to add to the service.

For example, to add the SSL proxy list *ssl list3* for a back-end SSL service, enter:

```
(config-service[server1])# add ssl-proxy-list ssllist3
```

To remove an SSL proxy list for the back-end service, enter:

```
(config-service[server1])# remove ssl-proxy-list ssllist3
```

Configuring the Back-End Service Keepalive Type

A service of type **ssl-accel-backend** supports the use of keepalives to periodically check the health of the SSL server. The CSS sends the keepalives to the IP address configured on the service.

To configure a keepalive, use the **keepalive type** command in service configuration mode. The syntax for this service configuration mode command is:

```
(config-service[server1])# keepalive type type
```

For the *type* variable, enter one of the following keepalive types:

- **icmp** - An ICMP echo message (ping). This is the default keepalive type.
- **none** - Do not send keepalive messages to a service.
- **ssl** - SSL HELLO keepalives for this service. The CSS sends a client HELLO to connect the SSL server. After the CSS receives a HELLO from the server, the CSS closes the connection with a TCP RST.
- **tcp** - A TCP session that determines service viability through a 3-way handshake and reset; SYN, SYN-ACK, ACK, RST-ACK.
- **http {non-persistent} encrypt** - Encrypted persistent or non-persistent HTTP keepalive to verify the full SSL handshake and data returned from the server.



Note

If you configure a TCP, SSL, or encrypted keepalive, you must configure the keepalive port correctly for the service to work.

For more information about ICMP, SSL, TCP, and other CSS keepalives, refer to the *Cisco Content Services Switch Content Load-Balancing Configuration Guide*.

For information about encrypted HTTP keepalives, see the following section.

Configuring Encrypted HTTP Keepalives

Encrypted HTTP keepalives allow the verification of the full SSL handshake and the data returned from the server. For a back-end SSL server, the keepalives perform an HTTP GET or HEAD that is broadcasted to all SSL modules in the CSS. The keepalive picks an SSL module and sends keepalive messages to the module until a keepalive failure occurs. Then the keepalive selects another SSL module in the CSS.



Note

An SSL proxy list contains a maximum of 256 SSL back-end or initiation servers. Therefore, the total number of encrypted keepalives on the CSS can only be 256.



Note

When configuring an encrypted HTTP keepalive, make sure that the configured IP address for the back-end server and the real server are the same.

You can configure a service or global encrypted keepalive. The syntax for the service configuration mode command is:

```
(config-service)# keepalive type http {non-persistent} encrypt
```

For example, to configure a service encrypted HTTP HEAD non-persistent keepalive for a back-end service, enter:

```
(config-service [ssl_serv1])# keepalive http non-persistent encrypt
```

The syntax for the global encrypted keepalive is:

```
(config-keepalive)# type http {non-persistent} encrypt
```

For global keepalives, no service or slot information is available to the keepalive. The keepalive selection of the module is dependent on the configuration of the service type. If all services are back-end services with the `ssl-accel-backend` service type, the keepalive can use all of the modules. If the configuration contains one SSL initiation service with the `ssl-init` service type, the keepalive uses the slot for that service.



Note

In order for encrypted global keepalives to work properly, you should only add it to services that use the same SSL proxy list.

For example, to configure a service encrypted HTTP HEAD non-persistent keepalive for a back-end service, enter:

```
(config-keepalive [SSL1])# http non-persistent encrypt
```

Then assign the global keepalive to the service for the back-end service.

If you are configuring a back-end SSL server to support session ID reuse and you want to use encrypted keepalives to perform a full SSL handshake, you must configure an additional back-end SSL server. This second server would have the same configuration as the original back-end server except you would configure it with a different port number and a session cache timeout value. Then you can configure an encrypted keepalive with the port number of the second back-end server. The encrypted keepalives are sent to this server on the SSL module. The module connects to the actual SSL back-end server and the keepalive performs a full SSL handshake.

Encrypted Keepalive Configuration Examples

The following configuration shows an example of encrypted HTTP non-persistent keepalives for a back-end SSL server. The HTTP GET non-persistent keepalives with a destination IP address 10.10.10.10 and port 80 are sent to an SSL module. Because service s1 is a back-end SSL server, proxy list p1 is broadcasted to all SSL modules in the CSS. Thus, the encrypted keepalive can select any SSL module. The keepalive stays with one module until the first keepalive probe failure occurs, and then the keepalive switches to another SSL module in the CSS.

```
!***** SSL PROXY LIST *****
ssl-proxy-list p1
  backend-server 1
  backend-server 1 IP address 10.10.10.10
  backend-server 1 server-IP 10.10.10.10
  active

!***** SERVICE *****
service s1
  IP address 10.10.10.10
  port 80
  type ssl-accel-backend
  add ssl-proxy-list p1
  keepalive type http non-persistent encrypt
  keepalive method get
  keepalive uri "/index.html"
  active
```

As shown in the following example, for encrypted keepalives to use a different handshake than the back-end server to support session ID reuse, you need to configure another back-end server with an identical configuration as the original back-end server, except for the port number and the session cache timeout value. Then, configure the encrypted keepalive port to match the port used by the new back-end server.

The HTTP GET non-persistent keepalives with destination IP address 10.10.10.10 and port 81 are sent to an SSL module. The IP address and port matches the back-end server 2 in the proxy list p1. Note that the address and port information for back-end server 1 is the address and port to the actual back-end server. Thus, the SSL module still sends the keepalive probes to the actual server except it performs the full SSL handshake because the session cache timeout is configured to zero.

```
!***** SSL PROXY LIST *****
ssl-proxy-list p1
  backend-server 1
    backend-server 1 IP address 10.10.10.10
    backend-server 1 port 80
    backend-server 1 server-IP 10.10.10.10
    backend-server 1 server-port 443
    backend-server 1 session-cache 500
  backend-server 2
    backend-server 2 IP address 10.10.10.10
    backend-server 2 port 81
    backend-server 2 server-IP 10.10.10.10
    backend-server 2 server-port 443
    backend-server 2 session-cache 0
  active

!***** SERVICE *****
service s1
  IP address 10.10.10.10
  port 80
  type ssl-accel-backend
  add ssl-proxy-list p1
  keepalive type http non-persistent encrypt
  keepalive port 81
  keepalive method get
  keepalive uri "/index.html"
  active
```

Configuring an IP Address for a Back-End SSL Service

The IP address for a back-end SSL service must match the IP address configured in the SSL proxy list for the back-end server.

For example, to configure the IP address 10.11.21.13 for the back-end SSL service, enter:

```
(config-service[server1])# ip address 10.11.21.13
```

To remove the IP address for the back-end SSL service, enter:

```
(config-service[server1])# no ip address
```

Configuring the Port Number for a Back-End SSL Service

The CSS uses the port number to send clear text data back to the SSL module for reencryption. By default, the CSS uses the port number of the back-end content rule associated with the service, port 80. If the port number is different from the the back-end HTTP-SSL content rule, use the **port** command to configure it.

Enter the port number as a integer from 1 to 65535. If you configure a port number, it must match the virtual port number configured in the SSL proxy list for the back-end server.

For example, to configure a port number of 55, enter:

```
(config-service[server1])# port 55
```

To reset the port number of the back-end content rule, enter:

```
(config-service[server1])# no port
```

Activating the SSL Service

Once you configure an SSL proxy list service, use the **active** command to activate the service. Activating a service puts it into the resource pool for load-balancing SSL content requests between the client and the server.

Before activating an SSL service:

- For a virtual SSL server, you must add an SSL proxy list to an **ssl-accel** type service before you can activate the service. If no list is configured when you enter the **active** command, the CSS logs the following error message and does not activate the service.

```
Must add at least one ssl-proxy-list to an ssl-accel type service
```

- For a back-end SSL server, you must add an SSL proxy list to an **ssl-accel-backend** type service before you can activate the service. If no list is configured when you enter the **active** command, the CSS logs the following error message and does not activate the service.

```
Must add at least one ssl-proxy-list to an ssl-accel type service
```

- The SSL proxy list added to the service must be active before you can activate the service. If the list is suspended, the CSS logs the following error message and does not activate the service.

```
No ssl-lists on service, service not activated
```

Once the service is ready to activate, the CSS initiates the transfer of appropriate SSL configuration data for each SSL proxy list to a specific SSL module and activates the service. If there is an error in transfer, the CSS logs the appropriate error and does not activate the service.

To activate service *ssl_serv1*, enter:

```
(config-service[ssl_serv1])# active
```

Suspending the SSL Service

To suspend an SSL service and remove it from the pool for future load-balancing SSL content requests, use the **suspend** command. Suspending an SSL service does not affect existing content flows, but it prevents additional connections from accessing the service for its content.

To suspend service `ssl_serv1`, enter:

```
(config-service[ssl_serv1])# suspend
```

Configuring a Content Rule for Back-End SSL

For the CSS to direct SSL requests for content, apply the back-end services to content rules. No network traffic is sent to an SSL module until you activate an SSL content rule to define where the content physically resides, where to direct the request for content (which SSL service), and which load-balancing method to use.

For an HTTP server or back-end SSL server content rule, ensure that each VIP address and port configured in the rule matches a VIP address and port configured in the cipher suite parameter for a virtual SSL server entry in the SSL proxy list (see the “[Specifying Cipher Suites](#)” section).

For a back-end server, you can specify a Layer 5 cookie or URL rule. The information in the rule finds a sticky server to use or load balances a new server for a new client request.

For more information on Layer 5 sticky and content rules, refer to the *Cisco Content Services Switch Content Load-Balancing Configuration Guide*.

