Example SSL Configurations

This appendix has the following sections:

Policy-Based Routing Configuration Example

This section shows a policy-based routing configuration example using a real client and a real server.

In Figure A-1, the SSL Services Module and the real server both have the IP address 3.100.100.151. The IP address on the SSL Services Module is configured with the **secondary** keyword and will not reply to ARP requests for this address, which avoids the duplicate IP address issue.

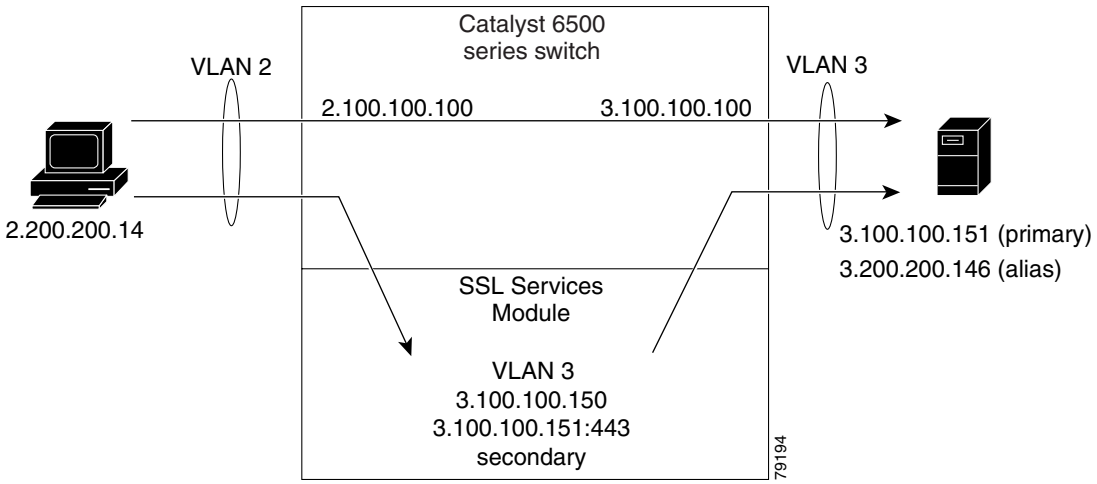
The client (2.200.200.14) is attached to a VLAN 2 switchport (access mode). The client's default gateway is 2.100.100.100 (VLAN 2 IP address on the supervisor engine).

The real server is attached to a VLAN 3 switchport (access mode). The default gateway on the real server is 3.100.100.100 (VLAN 3 IP address on the supervisor engine). The real server has two addresses: 3.100.100.151 (primary) and 3.200.200.146 (alias).

Clear-text (HTTP) traffic destined for 3.100.100.151 port 80 is sent directly to the real server, which bypasses the SSL Services Module.

With policy-based routing, SSL traffic destined for 3.100.100.151 port 443 is redirected to the SSL Services Module for decryption. The decrypted traffic is sent to 3.200.200.146 port 81 (the alias IP address for the real server). The return traffic from the real server is forwarded to the SSL Services Module. The module encrypts the traffic and sends it to client.

Figure A-1 Client-to-Server Traffic Flow Example



Configuring the Allowed VLANs

These examples show how to allow VLAN 3 between the SSL Services Module and the supervisor engine:

Cisco IOS Software

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ssl-proxy module 8 allowed-vlan 3
Router(config)# ^Z
Router#
Router# show ssl-proxy module 8 state
SSL-proxy module 8 data-port:
 Switchport:Enabled
Administrative Mode:trunk
Operational Mode:trunk
Administrative Trunking Encapsulation:dot1q
Operational Trunking Encapsulation:dot1q
Negotiation of Trunking:Off
Access Mode VLAN:1 (default)
Trunking Native Mode VLAN:1 (default)
Trunking VLANs Enabled:3
Pruning VLANs Enabled:2-1001
Vlans allowed on trunk:3
Vlans allowed and active in management domain:3
Vlans in spanning tree forwarding state and not pruned:
   3
Allowed-vlan :3

Router#
```

Catalyst Operating System Software

```
Console> (enable) set trunk 8/1 3
Adding vlans 3 to allowed list.
Console> (enable) show trunk 8/1
* - indicates vtp domain mismatch
# - indicates dot1q-all-tagged enabled on the port
Port      Mode         Encapsulation  Status        Native vlan
--------  -----------  -------------  ------------  -----------
 8/1      nonegotiate  dot1q          not-trunking  1

Port      Vlans allowed on trunk
--------  ---------------------------------------------------------------------
 8/1      3

Port      Vlans allowed and active in management domain
--------  ---------------------------------------------------------------------
 8/1      3

Port      Vlans in spanning tree forwarding state and not pruned
--------  ---------------------------------------------------------------------
 8/1      3
```

Configuring the Access List and Route Map

This example shows how to configure the access list and route map for redirecting SSL traffic from the client to the SSL Services Module and for redirecting clear text traffic from the real server to the SSL Services Module:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#
Router(config)# ip access-list extended redirect_ssl
Router(config-ext-nacl)# permit tcp any 3.0.0.0 0.255.255.255 eq 443
Router(config-ext-nacl)# !
Router(config-ext-nacl)# ip access-list extended reverse_traffic
Router(config-ext-nacl)# permit tcp 3.0.0.0 0.255.255.255 eq 81 any
Router(config-ext-nacl)# !
Router(config-ext-nacl)# route-map redirect_ssl permit
Router(config-route-map)# match ip address redirect_ssl
Router(config-route-map)# set ip next-hop 3.100.100.150
Router(config-route-map)# !
Router(config-route-map)# route-map reverse_traffic permit
Router(config-route-map)# match ip address reverse_traffic
Router(config-route-map)# set ip next-hop 3.100.100.150
Router(config-route-map)# !
Router(config-route-map)# interface Vlan2
Router(config-if)# ip address 2.100.100.100 255.0.0.0
Router(config-if)# ip policy route-map redirect_ssl
Router(config-if)# !
Router(config-if)# interface Vlan3
Router(config-if)# ip address 3.100.100.100 255.0.0.0
Router(config-if)# ip policy route-map reverse_traffic
Router(config-if)# !
Router(config-if)#^Z
Router#
```

Importing a Test Certificate

This example shows how to import the test certificate. For information on configuring a trustpoint and obtaining a certificate, see the "Configuring Keys and Certificates" section on page 3-3:

```
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 cisco"
% Then you can assign the trustpoint to a proxy service for testing.

*Oct  9 19:49:17.570:%STE-6-PKI_TEST_CERT_INSTALL:Test key and certificate was installed
into NVRAM in a PKCS#12 file.
ssl-proxy# configure terminal
ssl-proxy(config)# crypto ca import sample pkcs12 nvram: cisco
Source filename [sample]? test/testssl.p12
ssl-proxy(config)#
*Oct  9 19:51:04.674:%SSH-5-ENABLED:SSH 1.5 has been enabled
*Oct  9 19:51:04.678:%CRYPTO-6-PKCS12IMPORT_SUCCESS:PKCS #12 Successfully Imported.
ssl-proxy(config)# ^Z
ssl-proxy#
```

Configuring the SSL Proxy VLAN

This example shows how to add an interface to VLAN 3 on the SSL Services Module:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy vlan 3
ssl-proxy(config-vlan)# ipaddr 3.100.100.150 255.0.0.0
ssl-proxy(config-vlan)# gateway 3.100.100.100
ssl-proxy(config-vlan)# admin
ssl-proxy(config-vlan)# ^Z
ssl-proxy#
```

Configuring the SSL Proxy Service

This example shows how to add a specific proxy service that identifies a virtual IP address and a server IP address for each proxy:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy service sample
ssl-proxy(config-ssl-proxy)# virtual ipaddr 3.100.100.151 protocol tcp port 443 secondary
ssl-proxy(config-ssl-proxy)# server ipaddr 3.200.200.146 protocol tcp port 81
ssl-proxy(config-ssl-proxy)# cert rsa general-purpose trustpoint sample
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# ^Z
ssl-proxy#
```

Verifying Service and Connections

This example shows how to verify the SSL proxy service and connections:

```
ssl-proxy# show ssl-proxy service sample
Service id:3, bound_service_id:259
Virtual IP:3.100.100.151, port:443
Server IP:3.200.200.146, port:81
rsa-general-purpose certificate trustpoint:sample
  Certificate chain in use for new connections:
    Server Certificate:
       Key Label:sample
       Serial Number:01
    Root CA Certificate:
       Serial Number:00
  Certificate chain complete
Admin Status:up
Operation Status:up
ssl-proxy#

ssl-proxy# show ssl-proxy conn
Connections for TCP module 1
Local Address       Remote Address          VLAN   Conid  Send-Q Rwind  Recv-Q State
------------------- -------------------     -----  ------ ------ ------ ------ ------
3.100.100.151.443   2.200.200.14.37820      3      470     0     32768    0    ESTABLISHED
2.200.200.14.37820  3.200.200.146.81        3      471     0     32768    0    ESTABLISHED
ssl-proxy#
```

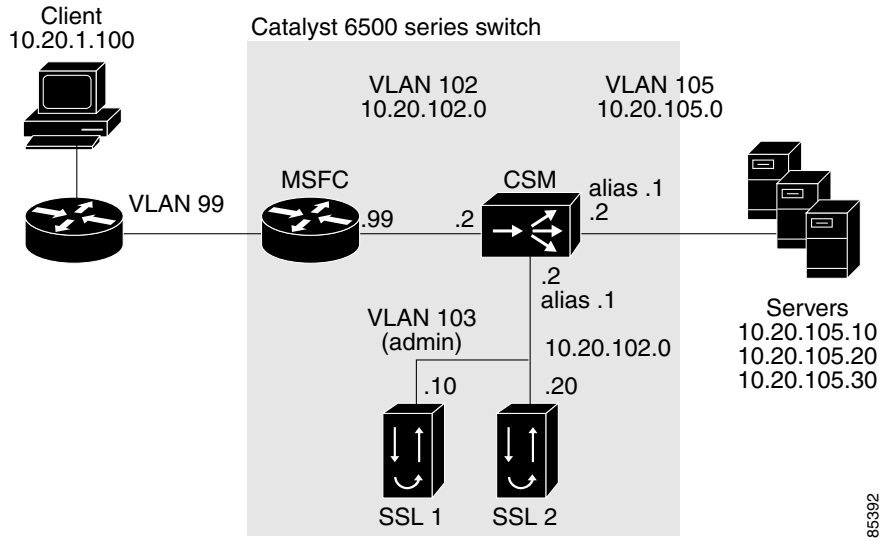
CSM and SSL Services Module Configuration Example (Bridge Mode, No NAT)

This section describes a CSM and SSL Services Module configuration that contains two SSL Services Modules, a CSM, a client network, and a server farm that has three web servers (IP addresses 10.20.105.10, 10.20.105.20, 10.20.105.30).

In this example, the CSM client VLAN and CSM server VLAN for the SSL Services Modules are configured in the same IP subnet (bridge mode), while the CSM server VLAN for the web servers is in a separate IP subnet. (See Figure A-2.)

The CSM is configured so that it does not perform NAT operations when it is load balancing encrypted traffic to the SSL Services Modules. The SSL Services Modules are also configured not to perform NAT operations when they are sending decrypted traffic back to the CSM. The CSM is then configured to perform NAT for the decrypted traffic to the selected destination server.

Figure A-2 Bridge Mode, No NAT Configuration Example



The following addresses are configured on the CSM virtual servers:

- Client clear text traffic—10.20.102.100:80
- Client SSL traffic—10.20.102.100:443
- Decrypted traffic from SSL Services Modules—10.20.102.100:80

The following address is configured on the SSL virtual server:

- 10.20.103.100:443 (this IP address is configured with the **secondary** keyword)

Figure A-2 shows VLAN 102 and VLAN 103 in the same subnet and VLAN 105 in a separate subnet.

Add all the required VLANs to the VLAN database, and configure the IP interface for VLAN 102 on the MSFC. Configure VLAN 102, VLAN 103, and VLAN105 on the CSM. See the "Initial SSL Services Module Configuration" section on page 2-2 for information on how to configure VLANs and IP interfaces.

Note While VLAN 102 exists as Layer 3 interface on the MSFC, both VLAN 103 and VLAN 105 exist only as VLANs in the VLAN database and as CSM VLANs, but they do not have corresponding Layer 3 interfaces on the MSFC.

This example shows how to create the client and server VLANs on the CSM installed in slot number 5:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module csm 5
Router(config-module-csm)# vlan 102 client
Router(config-slb-vlan-client)# ip address 10.20.102.2 255.255.255.0
Router(config-slb-vlan-client)# gateway 10.20.102.99
Router(config-slb-vlan-client)# exit
Router(config-module-csm)# vlan 103 server
Router(config-slb-vlan-server)# ip address 10.20.102.2 255.255.255.0
Router(config-slb-vlan-server)# alias 10.20.102.1 255.255.255.0
Router(config-slb-vlan-server)# exit
```

```
Router(config-module-csm)# vlan 105 server
Router(config-slb-vlan-server)# ip address 10.20.105.2 255.255.255.0
Router(config-slb-vlan-server)# alias 10.20.105.1 255.255.255.0
Router(config-slb-vlan-server)# end
```

This example shows how to allow VLAN 103 between the SSL Services Module and the CSM:

Cisco IOS Software

```
Router(config)# ssl-proxy module 4 allowed-vlan 103
```

Catalyst Operating System Software

```
Console> (enable) set trunk 4/1 103
```

This example shows how to create the server farm of web servers (configured with server NAT) and the server farm of SSL Services Modules (configured with no server NAT):

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module csm 5
Router(config-module-csm)# serverfarm SSLFARM
Router(config-slb-sfarm)# no nat server
Router(config-slb-sfarm)# real 10.20.102.10
Router(config-slb-real)# inservice
Router(config-slb-real)# real 10.20.102.20
Router(config-slb-real)# inservice
Router(config-slb-real)# exit
Router(config-slb-sfarm)# exit
Router(config-module-csm)# serverfarm WEBSERVERS
Router(config-slb-sfarm)# nat server
Router(config-slb-sfarm)# real 10.20.105.10
Router(config-slb-real)# inservice
Router(config-slb-real)# real 10.20.105.20
Router(config-slb-real)# inservice
Router(config-slb-real)# real 10.20.105.30
Router(config-slb-real)# inservice
Router(config-slb-real)# end
```

This example shows how to configure the three virtual servers. In this example, the web servers are receiving traffic to port 80 only, either directly from the clients or as decrypted traffic from the SSL Services Modules (since no port translation is configured).

The CSM distinguishes between requests received directly from the clients and requests received from the SSL Services Modules based on the VLAN from where the connections are received.

A sticky group is also configured to maintain stickiness based on the SSL ID.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module csm 5
Router(config-module-csm)# sticky 100 ssl timeout 30
Router(config-module-csm)# vserver CLEAR_VIP
Router(config-slb-vserver)# virtual 10.20.102.100 tcp www
Router(config-slb-vserver)# vlan 102
Router(config-slb-vserver)# serverfarm WEBSERVERS
Router(config-slb-vserver)# inservice
Router(config-slb-vserver)# exit
Router(config-module-csm)# vserver DECRYPT_VIP
Router(config-slb-vserver)# virtual 10.20.102.100 tcp www
Router(config-slb-vserver)# vlan 103
Router(config-slb-vserver)# serverfarm WEBSERVERS
Router(config-slb-vserver)# inservice
```

```
Router(config-slb-vserver)# exit
Router(config-module-csm)# vserver SSL_VIP
Router(config-slb-vserver)# virtual 10.20.102.100 tcp https
Router(config-slb-vserver)# vlan 102
Router(config-slb-vserver)# serverfarm SSLFARM
Router(config-slb-vserver)# sticky 30 group 100
Router(config-slb-vserver)# inservice
Router(config-slb-vserver)# end
```

This example shows how to configure the SSL Services Module to communicate with the CSM over VLAN 103, the admin VLAN:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy vlan 103
ssl-proxy(config-vlan)# ipaddr 10.20.102.10 255.255.255.0
ssl-proxy(config-vlan)# gateway 10.20.102.99
ssl-proxy(config-vlan)# admin
ssl-proxy(config-vlan)# end
```

To complete the configuration, enter the **ssl-proxy service** command to create a new service on the SSL Services Module (**test1**). This example shows how to configure a virtual IP address that matches the virtual server created on the CSM. (This virtual IP address is configured with the **secondary** keyword so that the SSL Services Module does not reply to ARP requests for this IP address.) The service is configured to send decrypted traffic back to the CSM without performing NAT.

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy service test1
ssl-proxy(config-ssl-proxy)# virtual ipaddr 10.20.102.100 protocol tcp port 443 secondary
ssl-proxy(config-ssl-proxy)# server ipaddr 10.20.102.1 protocol tcp port 80
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint testtp
ssl-proxy(config-ssl-proxy)# no nat server
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# end
```

The following examples show the output of the various **show** commands on the MSFC and CSM:

```
Router# show module csm 5 vlan detail
vlan   IP address       IP mask          type
-------------------------------------------------------
102    10.20.102.2      255.255.255.0    CLIENT
  GATEWAYS
  10.20.102.99
103    10.20.102.2      255.255.255.0    SERVER
  ALIASES
  IP address       IP mask
  -------------------------------
  10.20.102.1      255.255.255.0
105    10.20.105.2      255.255.255.0    SERVER
  ALIASES
  IP address       IP mask
  -------------------------------
  10.20.105.1      255.255.255.0

Router# show module csm 5 vserver detail
SSL_VIP, type = SLB, state = OPERATIONAL, v_index = 13
  virtual = 10.20.102.100/32:443, TCP, service = NONE, advertise = FALSE
  idle = 3600, replicate csrp = none, vlan = 102, pending = 30
  max parse len = 600, persist rebalance = TRUE
  conns = 0, total conns = 2
```

```
  Default policy:
    server farm = SSLFARM, backup = <not assigned>
    sticky: timer = 30, subnet = 0.0.0.0, group id = 100
  Policy          Tot Conn     Client pkts  Server pkts
  ------------------------------------------------------
  (default)       2            22           15

CLEAR_VIP, type = SLB, state = OPERATIONAL, v_index = 14
  virtual = 10.20.102.100/32:80, TCP, service = NONE, advertise = FALSE
  idle = 3600, replicate csrp = none, vlan = 102, pending = 30
  max parse len = 600, persist rebalance = TRUE
  conns = 0, total conns = 0
  Default policy:
    server farm = WEBSERVERS, backup = <not assigned>
    sticky: timer = 0, subnet = 0.0.0.0, group id = 0
  Policy          Tot Conn     Client pkts  Server pkts
  ------------------------------------------------------
  (default)       0            0            0

DECRYPT_VIP, type = SLB, state = OPERATIONAL, v_index = 15
  virtual = 10.20.102.100/32:80, TCP, service = NONE, advertise = FALSE
  idle = 3600, replicate csrp = none, vlan = 103, pending = 30
  max parse len = 600, persist rebalance = TRUE
  conns = 0, total conns = 2
  Default policy:
    server farm = WEBSERVERS, backup = <not assigned>
    sticky: timer = 0, subnet = 0.0.0.0, group id = 0
  Policy          Tot Conn     Client pkts  Server pkts
  ------------------------------------------------------
  (default)       2            11           7
```

The following examples show the output of the various **show** commands on the SSL Services Module:

```
ssl-proxy# show ssl-proxy service test1
Service id: 0, bound_service_id: 256
Virtual IP: 10.20.102.100, port: 443 (secondary configured)
Server IP: 10.20.102.1, port: 80
rsa-general-purpose certificate trustpoint: testtp
  Certificate chain in use for new connections:
    Server Certificate:
       Key Label: testtp
       Serial Number: 01
    Root CA Certificate:
      Serial Number: 00
  Certificate chain complete
Admin Status: up
Operation Status: up
ssl-proxy#
ssl-proxy# show ssl-proxy stats
TCP Statistics:
    Conns initiated   : 2            Conns accepted   : 2
    Conns established : 4            Conns dropped    : 4
    Conns closed      : 4            SYN timeouts     : 0
    Idle timeouts     : 0            Total pkts sent  : 26
    Data packets sent : 15           Data bytes sent  : 8177
    Total Pkts rcvd   : 27           Pkts rcvd in seq : 11
    Bytes rcvd in seq : 5142
```

```
SSL stats:
    conns attempted     : 2             conns completed     : 2
    full handshakes     : 2             resumed handshakes  : 0
    active conns        : 0             active sessions     : 0
    renegs attempted    : 0             conns in reneg      : 0
    handshake failures  : 0             data failures       : 0
    fatal alerts rcvd   : 0             fatal alerts sent   : 0
    no-cipher alerts    : 0             ver mismatch alerts : 0
    no-compress alerts  : 0             bad macs received   : 0
    pad errors          : 0

FDU Statistics
    IP Frag Drops     :  0               Serv_Id Drops     :  0
    Conn Id Drops     :  0               Checksum Drops    :  0
    IOS Congest Drops :  0               IP Version Drops  :  0
    Hash Full Drops   :  0               Hash Alloc Fails  :  0
    Flow Creates      :  4               Flow Deletes      :  4
    conn_id allocs    :  4               conn_id deallocs  :  4
    Tagged Drops      :  0               Non-Tagged Drops  :  0
    Add ipcs          :  0               Delete ipcs       :  0
    Disable ipcs      :  0               Enable ipcs       :  0
    Unsolicited ipcs  :  0               Duplicate ADD ipcs:  0
ssl-proxy#
```

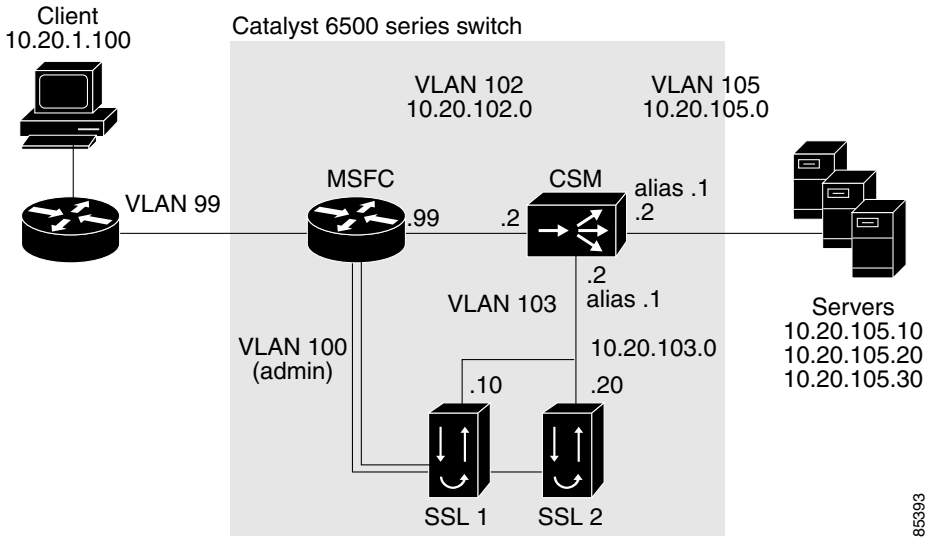
CSM and SSL Services Module Configuration Example (Router Mode, Server NAT)

This section describes a CSM and SSL Services Module configuration that contains two SSL Services Modules, a CSM, a client network, and a server farm that has three web servers (IP addresses 10.20.105.10, 10.20.105.20, 10.20.105.30).

In this example, the three CSM VLANs (client VLAN, server VLAN for the SSL Services Modules, and server VLAN for the web servers) are configured in distinct IP subnets (router mode). (See Figure A-3.)

The CSM is configured to perform server NAT operations when it is load balancing the encrypted traffic to the SSL Services Modules. The SSL Services Modules are also configured to perform server NAT operations when they are sending decrypted traffic back to the CSM. The CSM is then configured to perform NAT on the decrypted traffic to the selected destination server.

Figure A-3 Configuration Example—Router Mode, Server NAT



The following addresses are configured on the CSM virtual servers:

- Client clear text traffic—10.20.102.100:80
- Client SSL traffic—10.20.102.100:443
- Decrypted traffic from SSL Services Modules—10.20.103.100:81

The following addresses are configured on the SSL virtual server:

- 10.20.103.110:443
- 10.20.103.120:443

In Figure A-3, VLAN 102, VLAN 103, and VLAN 105 are in separate subnets. VLAN 100 (admin) is set up as a separate VLAN for management purposes.

Add all the required VLANs to the VLAN database, and configure the IP interfaces for VLAN 100 and VLAN 102 on the MSFC. Configure VLAN 102, VLAN 103, and VLAN 105 on the CSM. See the "Initial SSL Services Module Configuration" section on page 2-2 for information on how to configure VLANs and IP interfaces.

Note While VLAN 100 and VLAN 102 exist as Layer 3 interfaces on the MSFC, both VLAN 103 and VLAN 105 exist only as VLANs in the VLAN database and as CSM VLANs, but they do not have corresponding Layer 3 interfaces on the MSFC.

This example shows how to create the client and server VLANs on the CSM installed in slot number 5:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module csm 5
Router(config-module-csm)# vlan 102 client
Router(config-slb-vlan-client)# ip address 10.20.102.2 255.255.255.0
Router(config-slb-vlan-client)# alias 10.20.102.1 255.255.255.0
Router(config-slb-vlan-client)# gateway 10.20.102.99
Router(config-slb-vlan-client)# exit
Router(config-module-csm)# vlan 103 server
Router(config-slb-vlan-server)# ip address 10.20.103.2 255.255.255.0
Router(config-slb-vlan-server)# alias 10.20.103.1 255.255.255.0
```

```
Router(config-slb-vlan-server)# exit
Router(config-module-csm)# vlan 105 server
Router(config-slb-vlan-server)# ip address 10.20.105.2 255.255.255.0
Router(config-slb-vlan-server)# alias 10.20.105.1 255.255.255.0
Router(config-slb-vlan-server)# end
```

This example shows how to allow VLAN 103 (client VLAN) between the SSL Services Module and the CSM, and VLAN 100 (admin VLAN) between the SSL Services Module and the MSFC:

Cisco IOS Software

```
Router(config)# ssl-proxy module 4 allowed-vlan 100,103
```

Catalyst Operating System Software

```
Console> (enable) set trunk 4/1 100,103
```

This example shows how to create the server farm of web servers (configured with server NAT) and the server farm of SSL Services Modules (configured with server NAT):

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module csm 5
Router(config-module-csm)# serverfarm SSLFARM
Router(config-slb-sfarm)# nat server
Router(config-slb-sfarm)# real 10.20.103.110
Router(config-slb-real)# inservice
Router(config-slb-real)# real 10.20.103.120
Router(config-slb-real)# inservice
Router(config-slb-real)# exit
Router(config-slb-sfarm)# exit
Router(config-module-csm)# serverfarm WEBSERVERS
Router(config-slb-sfarm)# nat server
Router(config-slb-sfarm)# real 10.20.105.10
Router(config-slb-real)# inservice
Router(config-slb-real)# real 10.20.105.20
Router(config-slb-real)# inservice
Router(config-slb-real)# real 10.20.105.30
Router(config-slb-real)# inservice
Router(config-slb-real)# end
```

This example shows how to configure the three virtual servers. In this example, the web servers receive requests to port 80 directly from the clients, and decrypted requests to port 81 from the SSL Services Modules (since IP and port translation are configured).

This example also shows how to configure a sticky group to maintain stickiness based on the SSL ID:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module csm 5
Router(config-module-csm)# sticky 100 ssl timeout 30
Router(config-module-csm)# vserver CLEAR_VIP
Router(config-slb-vserver)# virtual 10.20.102.100 tcp www
Router(config-slb-vserver)# vlan 102
Router(config-slb-vserver)# serverfarm WEBSERVERS
Router(config-slb-vserver)# inservice
Router(config-slb-vserver)# exit
Router(config-module-csm)# vserver DECRYPT_VIP
Router(config-slb-vserver)# virtual 10.20.103.100 tcp 81
Router(config-slb-vserver)# vlan 103
Router(config-slb-vserver)# serverfarm WEBSERVERS
Router(config-slb-vserver)# inservice
Router(config-slb-vserver)# exit
```

```
Router(config-module-csm)# vserver SSL_VIP
Router(config-slb-vserver)# virtual 10.20.102.100 tcp https
Router(config-slb-vserver)# vlan 102
Router(config-slb-vserver)# serverfarm SSLFARM
Router(config-slb-vserver)# sticky 30 group 100
Router(config-slb-vserver)# inservice
Router(config-slb-vserver)# end
```

This example shows how to configure the SSL Services Module to communicate with the CSM over VLAN 103 and to communicate with the MSFC over VLAN 100 (admin VLAN):

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy vlan 103
ssl-proxy(config-vlan)# ipaddr 10.20.103.10 255.255.255.0
ssl-proxy(config-vlan)# gateway 10.20.103.1
ssl-proxy(config-vlan)# exit
ssl-proxy(config)# ssl-proxy vlan 100
ssl-proxy(config-vlan)# ipaddr 10.20.100.10 255.255.255.0
ssl-proxy(config-vlan)# gateway 10.20.100.99
ssl-proxy(config-vlan)# admin
ssl-proxy(config-vlan)# end
```

To complete the configuration, enter the **ssl-proxy service** command to create a new service on the SSL Services Module (**test1**). This example shows how to configure a virtual IP address, which acts as a real server for the CSM. (Since this virtual IP address is required to reply to ARP, the **secondary** keyword is not entered.) The service is configured to send decrypted traffic back to the CSM and to perform NAT on both the destination IP address and the port:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy service test1
ssl-proxy(config-ssl-proxy)# virtual ipaddr 10.20.103.110 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 10.20.103.100 protocol tcp port 81
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint testtp
ssl-proxy(config-ssl-proxy)# nat server
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# end
```

The following examples show the output of the various **show** commands on the MSFC and CSM:

```
Router# show mod csm 5 vlan detail
vlan   IP address       IP mask          type
-----------------------------------------------------
102    10.20.102.2      255.255.255.0    CLIENT
  GATEWAYS
  10.20.102.99
  ALIASES
  IP address       IP mask
  --------------------------------
  10.20.102.1      255.255.255.0
103    10.20.103.2      255.255.255.0    SERVER
  ALIASES
  IP address       IP mask
  --------------------------------
  10.20.103.1      255.255.255.0
105    10.20.105.2      255.255.255.0    SERVER
  ALIASES
  IP address       IP mask
  --------------------------------
  10.20.105.1      255.255.255.0
```

```
Router# show mod csm 5 vser detail
CLEAR_VIP, type = SLB, state = OPERATIONAL, v_index = 10
  virtual = 10.20.102.100/32:80, TCP, service = NONE, advertise = FALSE
  idle = 3600, replicate csrp = none, vlan = 102, pending = 30
  max parse len = 600, persist rebalance = TRUE
  conns = 0, total conns = 1
  Default policy:
    server farm = WEBSERVERS, backup = <not assigned>
    sticky: timer = 0, subnet = 0.0.0.0, group id = 0
  Policy          Tot Conn     Client pkts  Server pkts
  ------------------------------------------------------
  (default)       1            6            4

DECRYPT_VIP, type = SLB, state = OPERATIONAL, v_index = 11
  virtual = 10.20.103.100/32:81, TCP, service = NONE, advertise = FALSE
  idle = 3600, replicate csrp = none, vlan = 103, pending = 30
  max parse len = 600, persist rebalance = TRUE
  conns = 0, total conns = 2
  Default policy:
    server farm = WEBSERVERS, backup = <not assigned>
    sticky: timer = 0, subnet = 0.0.0.0, group id = 0
  Policy          Tot Conn     Client pkts  Server pkts
  ------------------------------------------------------
  (default)       2            11           7

SSL_VIP, type = SLB, state = OPERATIONAL, v_index = 13
  virtual = 10.20.102.100/32:443, TCP, service = NONE, advertise = FALSE
  idle = 3600, replicate csrp = none, vlan = 102, pending = 30
  max parse len = 600, persist rebalance = TRUE
  conns = 0, total conns = 2
  Default policy:
    server farm = SSLFARM, backup = <not assigned>
    sticky: timer = 30, subnet = 0.0.0.0, group id = 100
  Policy          Tot Conn     Client pkts  Server pkts
  ------------------------------------------------------
  (default)       2            21           15
```

The following examples show the output of the various **show** commands on the SSL Services Module:

```
ssl-proxy# show ssl-proxy service test1
Service id: 0, bound_service_id: 256
Virtual IP: 10.20.103.110, port: 443
Server IP: 10.20.103.100, port: 81
rsa-general-purpose certificate trustpoint: testtp
  Certificate chain in use for new connections:
    Server Certificate:
       Key Label: testtp
       Serial Number: 01
    Root CA Certificate:
      Serial Number: 00
  Certificate chain complete
Admin Status: up
Operation Status: up
ssl-proxy# show ssl-proxy stats
TCP Statistics:
    Conns initiated   : 2            Conns accepted   : 2
    Conns established : 4            Conns dropped    : 4
    Conns closed      : 4            SYN timeouts     : 0
    Idle timeouts     : 0            Total pkts sent  : 26
    Data packets sent : 15           Data bytes sent  : 8212
    Total Pkts rcvd   : 26           Pkts rcvd in seq : 11
    Bytes rcvd in seq : 5177
```

```
SSL stats:
    conns attempted     : 2             conns completed     : 2
    full handshakes     : 2             resumed handshakes  : 0
    active conns        : 0             active sessions     : 0
    renegs attempted    : 0             conns in reneg      : 0
    handshake failures  : 0             data failures       : 0
    fatal alerts rcvd   : 0             fatal alerts sent   : 0
    no-cipher alerts    : 0             ver mismatch alerts : 0
    no-compress alerts  : 0             bad macs received   : 0
    pad errors          : 0

FDU Statistics
    IP Frag Drops     :  0               Serv_Id Drops     :  0
    Conn Id Drops     :  0               Checksum Drops    :  0
    IOS Congest Drops :  0               IP Version Drops  :  0
    Hash Full Drops   :  0               Hash Alloc Fails  :  0
    Flow Creates      :  4               Flow Deletes      :  4
    conn_id allocs    :  4               conn_id deallocs  :  4
    Tagged Drops      :  0               Non-Tagged Drops  :  0
    Add ipcs          :  0               Delete ipcs       :  0
    Disable ipcs      :  0               Enable ipcs       :  0
    Unsolicited ipcs  :  0               Duplicate ADD ipcs:  0
```

Basic Backend Encryption Example

Backend encryption allows you to create a secure end-to-end environment. This example shows a basic backend encryption configuration.

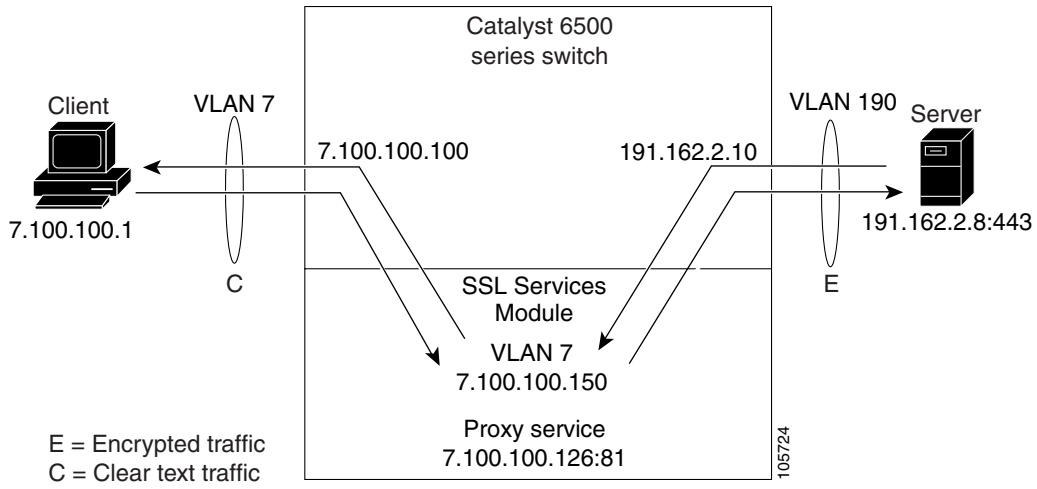
In Figure A-4, the client (7.100.100.1) is connected to switchport 6/47 in access VLAN 7. The server (191.162.2.8) is connected to switchport 10/2 in access VLAN 190.

The SSL proxy VLAN 7 has the following configuration:

- IP address—7.100.100.150
- Static route and gateway:
  - Route 191.0.0.0
  - Gateway 7.100.100.100

  The gateway IP address (the IP address of interface VLAN 7 on the MSFC) is configured so that the client-side traffic that is destined to an unknown network is forwarded to that IP address for further routing to the client.
- Client-side gateway—7.100.100.100 (the IP address of VLAN 7 configured on the MSFC)
- Virtual IP address of client proxy service—7.100.100.150:81
- Server IP address—191.162.2.8

Figure A-4 Basic Backend Encryption



Configuring VLANS and Switchports

These examples show how to create VLANs and assign ports to the respective VLANs:

Cisco IOS Software

```
ssl-proxy# configure terminal
ssl-proxy(config)# vlan 7
ssl-proxy(config-vlan)# exit
ssl-proxy(config)# vlan 190
ssl-proxy(config-vlan)# exit
ssl-proxy(config)# interface FastEthernet6/47
ssl-proxy(config-if)# switchport
ssl-proxy(config-if)# switchport access vlan 7
ssl-proxy(config-if)# switchport mode access
ssl-proxy(config-if)# exit
ssl-proxy(config)#
ssl-proxy(config)# interface GigabitEthernet10/2
ssl-proxy(config-if)# switchport
ssl-proxy(config-if)# switchport access vlan 190
ssl-proxy(config-if)# switchport mode access
ssl-proxy(config-if)# exit
ssl-proxy(config)#
```

Catalyst Operating System

```
Console> (enable) set vlan 7
VTP advertisements transmitting temporarily stopped,
and will resume after the command finishes.
Vlan 7 configuration successful
Console> (enable)
Console> (enable) set vlan 190
VTP advertisements transmitting temporarily stopped,
and will resume after the command finishes.
Vlan 190 configuration successful
Console> (enable)
```

```
Console> (enable) set vlan 7 6/47
VLAN  Mod/Ports
---- ----------------------
7     6/47
Console> (enable) set vlan 190 10/2
VLAN  Mod/Ports
---- ----------------------
190   10/2
Console> (enable)
```

Configuring the Allowed VLANs

This example shows how to allow VLAN 7 between SSL module in slot 12 and the supervisor engine:

Cisco IOS Software

```
ssl-proxy# configure terminal
ssl-proxy(config)# ssl-proxy module 12 allowed-vlan 7
ssl-proxy(config)# exit
ssl-proxy#

ssl-proxy# show ssl-proxy mod 12 state
SSL-proxy module 12 data-port:

Switchport:Enabled
Administrative Mode:trunk
Operational Mode:trunk
Administrative Trunking Encapsulation:dot1q
Operational Trunking Encapsulation:dot1q
Negotiation of Trunking:Off
Access Mode VLAN:1 (default)
Trunking Native Mode VLAN:1 (default)
Trunking VLANs Enabled:7
Pruning VLANs Enabled:2-1001
Vlans allowed on trunk:7
Vlans allowed and active in management domain:7
Vlans in spanning tree forwarding state and not pruned:
   7
Allowed-vlan :7

ssl-proxy#
```

Catalyst Operating System

```
Console> (enable) show mod 12
Mod Slot Ports Module-Type               Model               Sub Status
--- ---- ----- ------------------------- ------------------- --- --------
12  12   1     SSL Module                WS-SVC-SSL-1        no  ok

Mod Module-Name          Serial-Num
--- -------------------- -----------
12                       SAD062004N0

Mod MAC-Address(es)                        Hw     Fw          Sw
--- -------------------------------------- ------ ----------- ------------------
12  00-e0-b0-ff-f0-c2                      0.305  7.2(1)      2.1(1)
Console> (enable)
Console> (enable) set trunk 12/1 7
Adding vlans 7 to allowed list.
Port(s) 12/1 allowed vlans modified to 7.
Console> (enable)
```

```
Console> (enable) show trunk 12/1
* - indicates vtp domain mismatch
# - indicates dot1q-all-tagged enabled on the port
$ - indicates non-default dot1q-ethertype value
Port      Mode         Encapsulation  Status        Native vlan
--------  -----------  -------------  ------------  -----------
12/1      nonegotiate  dot1q          trunking      1

Port      Vlans allowed on trunk
--------  ---------------------------------------------------------------------
12/1      7,190

Port      Vlans allowed and active in management domain
--------  ---------------------------------------------------------------------
12/1      7,190

Port      Vlans in spanning tree forwarding state and not pruned
--------  ---------------------------------------------------------------------
12/1      7,190
Console> (enable)
```

Configuring the Access List and Route Map

This example shows how to configure the access list and route map for redirecting SSL traffic from the server to the SSL Services Module and for redirecting clear text traffic from the client to the SSL Services Module:

```
ssl-proxy(config)# ip access-list extended client
ssl-proxy(config-ext-nacl)# permit tcp any host 7.100.100.126 eq 81
ssl-proxy(config-ext-nacl)# exit
ssl-proxy(config)#
ssl-proxy(config)# ip access-list extended server
ssl-proxy(config-ext-nacl)# permit tcp host 191.162.2.8 eq 443 any
ssl-proxy(config-ext-nacl)# exit
ssl-proxy(config)#
ssl-proxy(config)# route-map server permit 10
ssl-proxy(config-route-map)# match ip address server
ssl-proxy(config-route-map)# set ip next-hop 7.100.100.150
ssl-proxy(config-route-map)# exit
ssl-proxy(config)#
ssl-proxy(config)# route-map client permit 10
ssl-proxy(config-route-map)# match ip address client
ssl-proxy(config-route-map)# set ip next-hop 7.100.100.150
ssl-proxy(config-route-map)# exit
ssl-proxy(config)#
ssl-proxy(config)# interface Vlan7
ssl-proxy(config-if)# ip address 7.100.100.100 255.0.0.0
ssl-proxy(config-if)# ip policy route-map client
ssl-proxy(config-if)# end
ssl-proxy#
ssl-proxy# configure terminal
ssl-proxy(config)# interface Vlan190
ssl-proxy(config-if)# ip address 191.162.2.10 255.0.0.0
ssl-proxy(config-if)# ip policy route-map server
ssl-proxy(config-if)# end
```

Configuring the SSL Proxy VLAN

This example shows how to add an interface to VLAN 7 on the SSL Services Module:

```
ssl-module# configure terminal
ssl-module(config)# ssl-proxy vlan 7
ssl-module(config-vlan)# ipaddr 7.100.100.150 255.0.0.0
ssl-module(config-vlan)# gateway 7.100.100.100
ssl-module(config-vlan)# route 191.0.0.0 255.0.0.0 gateway 7.100.100.100
ssl-module(config-vlan)# exit
```

Configuring the Root Certificate Authority Trustpoint for Server Certificate Authentication

This example shows how to configure root certificate authority trustpoint for authenticating server certificates. See the "Server Certificate Authentication" section on page 3-48 for information on configuring server certificate authentication.

```
ssl-proxy(config)# crypto ca trustpoint root
ssl-proxy(ca-trustpoint)# enrollment terminal
ssl-proxy(ca-trustpoint)# exit
ssl-proxy(config)# crypto ca auth root

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIICSTCCAfOgAwIBAgIBATANBgkqhkiG9w0BAQUFADBxMQswCQYDVQQGEwJVUzET
MBEGA1UECBMKQ2FsaWZvcm5pYTERMA8GA1UEBxMIU2FuIEpvc2UxDjAMBgNVBAoT
BUNpc2NvMQwwCgYDVQQLEwNIU1MxHDAaBgNVBAMTE0NlcnRpZmljYXRlIE1hbmFn
ZXIwHhcNMDIwNDI3MDcwMDAwWhcNMDYwNDI3MDcwMDAwWjBxMQswCQYDVQQGEwJV
UzETMBEGA1UECBMKQ2FsaWZvcm5pYTERMA8GA1UEBxMIU2FuIEpvc2UxDjAMBgNV
BAoTBUNpc2NvMQwwCgYDVQQLEwNIU1MxHDAaBgNVBAMTE0NlcnRpZmljYXRlIE1h
bmFnZXIwXDANBgkqhkiG9w0BAQEFAANLADBIAkEAvadjzN/Z3SEe7dBktvG6U1/W
hoe1/vJ6uIV/goS/fwYnr0+Gk1Lw2DGEMN9P1li5qcM2Rgq6E+6AQv2UYerBjQID
AQABo3YwdDARBglghkgBhvhCAQEEBAMCAAcwDwYDVR0TAQH/BAUwAwEB/zAdBgNV
HQ4EFgQU7u9bvU3N9Wtgnc9GwuoleyKlCAAwHwYDVR0jBBgwFoAU7u9bvU3N9Wtg
nc9GwuoleyKlCAAwDgYDVR0PAQH/BAQDAgGGMA0GCSqGSIb3DQEBBQUAA0EAhYZX
upIv+ZRo639io4ZLaiCWePHA+6Oz2n4B1kX9Jqx9fS3Zbyc712BP6lM2Apk5fhBs
6z/WrDRR0GZhlOoAvg==
-----END CERTIFICATE-----
quit
Certificate has the following attributes:
Fingerprint:683F909E 0B9F1651 7AAB8E36 14DBE45F
% Do you accept this certificate? [yes/no]:yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy pool ca root-ca
ssl-proxy(config-ca-pool)# ca trustpoint root
ssl-proxy(config-ca-pool)# exit
```

Configuring the SSL Proxy Service

This example shows how to configure the SSL client proxy service to accept clear text connections to virtual IP address 7.100.100.126 with TCP port 81, and to initiate an SSL connection to the backend SSL server IP address 191.162.2.8 with destination TCP port 443. See the "SSL Client Proxy Services" section on page 3-42 for information on configuring client proxy services.

```
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy service backend-ssl client
ssl-proxy(config-ssl-proxy)# virtual ipaddr 7.100.100.126 protocol tcp port 81
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# trusted-ca root-ca
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# ^Z
ssl-proxy#
```

Verifying Service and Connections

This example shows the successful initiation of the SSL connection to the backend SSL server:

```
ssl-proxy# show ssl-proxy stats ssl
SSL Statistics:
    conns attempted     :5              conns completed     :5
    conns in handshake  :0              conns in data       :0
    renegs attempted    :0              conns in reneg      :0
    active sessions     :0              max handshake conns :1
    rand bufs allocated :1              cached rand buf miss:0
    current device q len:0              max device q len    :1
    sslv2 forwards      :0              cert reqs processed :0
    fatal alerts rcvd   :0              fatal alerts sent   :0
    stale packet drops  :0              service_id discards :0
    session reuses      :0              hs handle in use    :0

    SSL3 Statistics:
      full handshakes    :0              resumed handshakes :0
      handshake failures :0              data failures      :0
      bad macs received  :0              pad errors         :0
      conns established with cipher rsa-with-rc4-128-md5        :0
      conns established with cipher rsa-with-rc4-128-sha        :0
      conns established with cipher rsa-with-des-cbc-sha        :0
      conns established with cipher rsa-with-3des-ede-cbc-sha   :0

    TLS1 Statistics:
      full handshakes    :3              resumed handshakes :2
      handshake failures :0              data failures      :0
      bad macs received  :0              pad errors         :0
      conns established with cipher rsa-with-rc4-128-md5        :5
      conns established with cipher rsa-with-rc4-128-sha        :0
      conns established with cipher rsa-with-des-cbc-sha        :0
      conns established with cipher rsa-with-3des-ede-cbc-sha   :0

SSL error statistics:
    session alloc fails :0              session limit exceed:0
    handshake init fails:0              renegotiation fails :0
    no-cipher alerts    :0              ver mismatch alerts :0
    no-compress alerts  :0              multi buf rec errors:0
    ssl peer closes     :0              non-ssl peer closes :0
    unexpected record   :0              rec formatting error:0
    rsa pkcs pad errors :0              premaster errors    :0
    failed rsa reqs     :0              failed random reqs  :0
```

```
    failed key-material :0            failed master-secret:0
    failed update hash  :0            failed finish hash  :0
    failed encrypts     :0            failed decrypts     :0
    bad record version  :0            bad record size     :0
    cert verify errors  :0            unsupported certs   :0
    conn aborted        :0
    overload drops      :0            hs limit exceeded   :0
    hs handle mem fails :0            conn reuse error    :0
    dev invalid params  :0            dev failed requests :0
    dev timeout         :0            dev busy            :0
    dev cancelled       :0            no dev fails        :0
    dev resource fails  :0            dev unknown errors  :0
    dev conn ctx fails  :0            dev cmd ctx fails   :0
    mem alloc fails     :0            buf alloc fails     :0
    invalid cipher algo :0            invalid hash algo   :0
    unaligned buf addr  :0            unaligned buf len   :0
    internal error      :0            unknown ipcs        :0
    double free attempts:0            alert-send fails    :0

SSL Crypto Statistics:
    blocks encrypted    :20           blocks decrypted    :249
    bytes encrypted     :4898         bytes decrypted     :25194
    rsa public key ops  :7            rsa private key ops :4
    crypto failures     :0            device dma errors   :0

SSL last 5 sec average Statistics:
    full handshakes     :0            resumed handshakes  :0
    handshake failures  :0            data failures       :0
    bytes encrypted     :0            bytes decrypted     :0

SSL last 1 min average Statistics:
    full handshakes     :0            resumed handshakes  :0
    handshake failures  :0            data failures       :0
    bytes encrypted     :0            bytes decrypted     :0

SSL last 5 min average Statistics:
    full handshakes     :0            resumed handshakes  :0
    handshake failures  :0            data failures       :0
    bytes encrypted     :0            bytes decrypted     :0

SSL PKI Statistics:
    number of malloc    :224          number of free      :209
    ssl buf allocated   :12           ssl buf freed       :8

    Peer Certificate Verify Statistics:
    cert approved       :3            cert disapproved    :0
    peer cert empty     :0            total num of request:3
    req being processed :0            req pending         :0
    longest queue       :1            longest pending     :0
    verify congestion   :0            req dropped, q full :0
    no memory for verify:0            verify data error   :0
    verify context error:0            context delete error:0
    timer expired error :0            timer expired count :0
    late verify result  :0            timer turned on     :3
    timer turned off    :3            context created     :3
    context deleted     :3
```

```
    High Priority IPC:
    ipc request received:23            ipc request dropped :0
    ipc req duplicated  :0             ipc req fragment err:0
    ipc req parm len err:0             ipc req op code err :0
    ipc req cert len err:0             ipc response sent   :23
    ipc resp no memory  :0             ipc resp no ssl buf :0
    ipc buffer allocated:0             ipc buffer freed    :0
    ipc buf alloc failed:0             ipc send msg failed :0

    Normal Priority IPC:
    ipc buffer allocated:3             ipc buffer freed    :3
    ipc request sent    :3             ipc request received:3
    ipc buf alloc failed:0             ipc send msg failed :0
    ipc requests dropped:0

ssl-proxy#
```

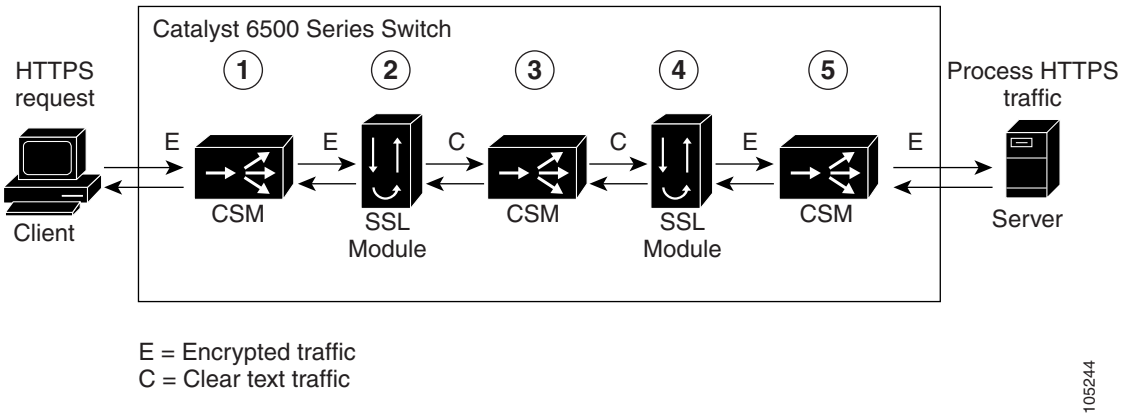
Integrated Secure Content-Switching Service Example

Configuring an integrated secure content-switching service (using a content switching module [CSM] as a server load balancer) with backend encryption has all the benefits of load-balancing and content switching, while securing data with full SSL coverage as it traverses paths of vulnerability.

As shown in Figure A-5, an integrated secure content-switching service configuration involves five processing steps:

1. The CSM load-balances the SSL traffic, based on either load-balancing rules or using the SSL sticky feature (see the "Sticky Connections" section on page 5-7 for information on configuring sticky connections), to an SSL Services Module.
2. The SSL Services Module terminates the SSL session, decrypts the SSL traffic into clear text traffic, and forwards the traffic back to the CSM.
3. The CSM content-switches the clear text traffic to the SSL Services Module again for encryption to SSL traffic.
4. The SSL Services Module forwards the encrypted SSL traffic to the CSM.
5. The CSM forwards the SSL traffic to the HTTPS server.

Figure A-5 Backend Encryption Example—Integrated Secure Content-Switching Service



Configuring the CSM

This example shows how to configure the VLANs on the CSM. VLAN 24 is the VLAN through which client traffic arrives. VLAN 35 is the VLAN between the SSL Services Module and the CSM.

```
Router# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# module ContentSwitchingModule 6
Router(config-module-csm)# vlan 24 client
Route(config-slb-vlan-client)# ip address 24.24.24.24 255.0.0.0
Route(config-slb-vlan-client)# vlan 35 server
Route(config-slb-vlan-server)# ip address 35.35.35.35 255.0.0.0
Route(config-slb-vlan-server)# route 36.0.0.0 255.0.0.0 gateway 35.200.200.3
```

This example shows how to configure the URL policy for Layer 7 parsing:

```
Route(config-slb-vlan-server)# map URL url
Router(config-slb-map-url)# match protocol http method GET url /*
```

This example shows how to create server farms:

```
Router(config-slb-map-url)# serverfarm SSLCARDS
Router(config-slb-sfarm)# real 35.200.200.101
Router(config-slb-real)# inservice

Router(config-slb-real)# serverfarm VLAN36REALS
Router(config-slb-sfarm)# real 36.200.200.14
Router(config-slb-real)# inservice
Router(config-slb-real)# real 36.200.200.5
Router(config-slb-real)# inservice
```

This example shows how to create the virtual servers:

```
Router(config-slb-real)# vserver LB-HTTP-SSLMODS
Router(config-slb-vserver)# virtual 35.35.35.25 tcp 81
Router(config-slb-vserver)# vlan 35
Router(config-slb-vserver)# slb-policy URL
Router(config-slb-vserver)# inservice

Router(config-slb-vserver)# vserver LB-SSL-SSLMODS
Router(config-slb-vserver)# virtual 24.24.24.25 tcp https
Router(config-slb-vserver)# serverfarm SSLCARDS
Router(config-slb-vserver)# inservice
```

This example shows how to display the status of the real servers and virtual servers:

```
Router# sh module contentSwitchingModule all reals
---------------------- CSM in slot 6 ----------------------

real                  server farm      weight  state          conns/hits
------------------------------------------------------------------------
35.200.200.101        SSLCARDS         8       OPERATIONAL    0
36.200.200.14         VLAN36REALS      8       OPERATIONAL    0
36.200.200.5          VLAN36REALS      8       OPERATIONAL    0

Router# sh module contentSwitchingModule all vservers
---------------------- CSM in slot 6 ----------------------

vserver         type  prot virtual                 vlan state        conns
------------------------------------------------------------------------
LB-HTTP-SSLMODS SLB   TCP  35.35.35.25/32:81       35   OPERATIONAL  0
LB-SSL-SSLMODS  SLB   TCP  24.24.24.25/32:443      ALL  OPERATIONAL  0

Router#
```

Configuring the SSL Services Module

This example shows how to create the VLAN between the SSL Services Module and the CSM:

```
ssl-proxy(config)# ssl-proxy vlan 35
ssl-proxy(config-vlan)# ipaddr 35.200.200.3 255.0.0.0
ssl-proxy(config-vlan)# gateway 35.200.200.100
ssl-proxy(config-vlan)# admin
```

This example shows how to configure a trusted certificate authority pool on the SSL Services Module:

```
ssl-proxy(config-vlan)# ssl-proxy pool ca net
ssl-proxy(config-ca-pool)# ca trustpoint keon-root
ssl-proxy(config-ca-pool)# ca trustpoint net-root
ssl-proxy(config-ca-pool)# ca trustpoint TP-1024-pcks12-root
```

This example shows how to configure a URL rewrite policy on the SSL Services Module:

```
ssl-proxy(config)# ssl-proxy policy url-rewrite frontend
ss(config-url-rewrite-policy)# url www.cisco.com clearport 80 sslport 443
ss(config-url-rewrite-policy)# url wwwin.cisco.com clearport 80 sslport 443
ss(config-url-rewrite-policy)# url wwwin.cisco.com clearport 81 sslport 443
```

This example shows how to configure the SSL server proxy that accepts client traffic coming through the CSM. This example also shows how to configure client authentication, SSL v2.0 forwarding, and URL rewrite policy.

Note For SSL V2.0 connections, the SSL Services Module directly opens a connection to SSL Services Module instead of giving it back to CSM.

```
ssl-proxy(config-ca-pool)# ssl-proxy service frontend
ssl-proxy(config-ssl-proxy)# virtual ipaddr 35.200.200.101 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 35.35.35.25 protocol tcp port 81
ssl-proxy(config-ssl-proxy)# server ipaddr 35.200.200.14 protocol tcp port 443 sslv2
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint TP-1024-pkcs12
ssl-proxy(config-ssl-proxy)# policy url-rewrite frontend
ssl-proxy(config-ssl-proxy)# trusted-ca net
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
```

This example shows how to configure the SSL client proxy that accepts clear text traffic from the CSM after the traffic completes Layer 7 parsing and decides the real server. This example also shows how to configure client certificates and a wildcard proxy.

Note The gateway address (35.200.200.125) is the address through which the real servers (36.200.200.14 and 36.200.200.5) are reached.

```
ssl-proxy(config-ssl-proxy)# ssl-proxy service wildcard client
ssl-proxy(config-ssl-proxy)# virtual ipaddr 0.0.0.0 0.0.0.0 protocol tcp port 81 secondary
ssl-proxy(config-ssl-proxy)# server ipaddr 35.200.200.125 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint client-cert
ssl-proxy(config-ssl-proxy)# no nat server
ssl-proxy(config-ssl-proxy)# trusted-ca net
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# ^Z
```

This example shows how to display the status of the SSL server proxy service:

```
ssl-proxy# show ssl-proxy service frontend
Service id: 2, bound_service_id: 258
Virtual IP: 35.200.200.101, port: 443
Server IP: 35.35.35.25, port: 81
SSLv2  IP: 35.200.200.14, port: 443
URL Rewrite Policy: frontend
Certificate authority pool: net
  CA pool complete
rsa-general-purpose certificate trustpoint: TP-1024-pkcs12
  Certificate chain for new connections:
    Certificate:
       Key Label: TP-1024-pkcs12, 1024-bit, not exportable
       Key Timestamp: 22:53:16 UTC Mar 14 2003
       Serial Number: 3C2CD2330001000000DB
    Root CA Certificate:
       Serial Number: 313AD6510D25ABAE4626E96305511AC4
  Certificate chain complete
Certificate authentication type: All attributes (like CRL) are verified
Admin Status: up
Operation Status: up
ssl-proxy#
```

This example shows how to display status of the SSL client proxy service:

```
ssl-proxy# show ssl-proxy service wildcard
Service id: 267, bound_service_id: 11
Virtual IP: 0.0.0.0, port: 81 (secondary configured)
Virtual IP mask: 0.0.0.0
Server IP: 35.200.200.125, port: 443
Certificate authority pool: net
  CA pool complete
rsa-general-purpose certificate trustpoint: client-cert
  Certificate chain for new connections:
    Certificate:
       Key Label: client-cert, 1024-bit, not exportable
       Key Timestamp: 18:42:01 UTC Jul 14 2003
       Serial Number: 04
    Root CA Certificate:
       Serial Number: 01
  Certificate chain complete
Certificate authentication type: All attributes (like CRL) are verified
Admin Status: up
Operation Status: up
ssl-proxy#
```

Site-To-Site Transport Layer VPN Example

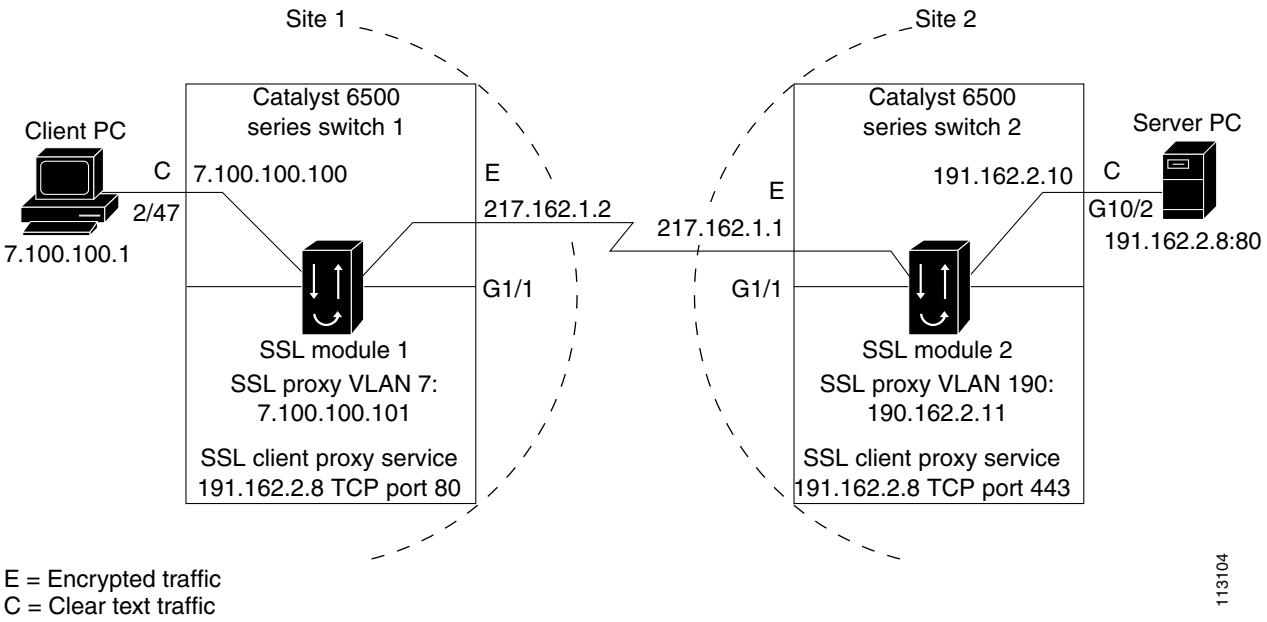
A site-to-site transport layer VPN configuration is used to connect two trusted sites to support TCP-based applications.

In Figure A-6, SSL module 1 is configured with a client proxy service. SSL module 1 encrypts the client clear text traffic into SSL traffic and forwards the encrypted SSL traffic to SSL module 2 on the remote site through a backend SSL session. SSL module 2 is configured with a standard SSL offload virtual service, which decrypts the received SSL traffic into clear text and forwards it to the servers on the remote site.

When you configure a proxy service as either clear text-to-encryption or encryption-to-clear text mode, the proxy service acts in an SSL client role while communicating with the secure backend server. You need to configure SSL policies to describe the SSL client and the backend SSL session. See the "Configuring SSL Policy" section on page 4-2 for information on configuring SSL policies.

This section gives an example of how to tunnel HTTP traffic from the client to the server and back to the client through an SSL VPN.

Figure A-6 Backend Encryption Example—Site-to-Site Transport Layer VPN



In Figure A-6, Site 1 and Site 2 are connected by Gigabit Ethernet; however, both sites could also be connected through the Internet.

The client PC (7.100.100.1) is connected to switchport 2/47 in access VLAN 7. The server (191.162.2.8) is connected to switchport 10/2 in access VLAN 190.

Site 1 Configuration

Site 1 in Figure A-6 shows the SSL Services Module (SSL module 1) installed in slot 13 in Catalyst 6500 series switch 1.

The following example shows how to add a VLAN between the SSL Services Module and the supervisor engine:

```
cat6k-router-1# show mod 13
Mod Ports Card Type                              Model              Serial No.
--- ----- -------------------------------------- ------------------ -----------
 13     1  SSL Module                             WS-SVC-SSL-1       SAD062503FZ

Mod MAC addresses                       Hw     Fw           Sw           Status
--- ----------------------------------- ------ ------------ ------------ -------
 13  0010.7b00.0e00 to 0010.7b00.0e07   0.304 7.2(1)       2.1(1)       Ok

Mod Online Diag Status
--- -------------------
 13 Pass
```

```
cat6k-router-1# config t
cat6k-router-1(config)# ssl-proxy module 13 allowed-vlan 7
cat6k-router-1(config)# exit
```

The following example shows to configure the VLAN, configure a port as a switchport, and assign the switchport to the access VLAN:

```
cat6k-router-1# config t
cat6k-router-1(config)# vlan 7
cat6k-router-1(config-vlan)# exit
cat6k-router-1(config)# interface FastEthernet2/47
cat6k-router-1(config-if)# switchport
cat6k-router-1(config-if)# switchport access vlan 7
cat6k-router-1(config-if)# switchport mode access
cat6k-router-1(config-if)# exit
cat6k-router-1(config)#
```

The following examples show how to configure extended access lists:

- Access list "client" is used to match any traffic going to IP address 191.162.2.8 with destination TCP port 80 (HTTP traffic).

  ```
  cat6k-router-1(config)# ip access-list extended client
  cat6k-router(config-ext-nacl)# permit tcp any host 191.162.2.8 eq www
  cat6k-router(config-ext-nacl)# exit
  ```

- Access list "server" is used to match any traffic from IP address 191.162.2.8 with source TCP port 443 (encrypted traffic from site 2).

  ```
  cat6k-router-1(config)# ip access-list extended server
  cat6k-router(config-ext-nacl)# permit tcp host 191.162.2.8 eq 443 any
  cat6k-router(config-ext-nacl)# exit
  ```

The following examples show how to configure route maps to redirect traffic to the SSL Services Module for encryption and decryption:

- Route map "client" redirects the traffic that matches access-list "client" to the next hop IP address 7.100.100.101 (the IP address of SSL proxy VLAN 7 on SSL-module-1).

  ```
  cat6k-router-1(config)# route-map client permit 10
  cat6k-route(config-route-map)# match ip address client
  cat6k-route(config-route-map)# set ip next-hop 7.100.100.101
  cat6k-route(config-route-map)# exit
  ```

- Route map "server" redirects the traffic that matches access-list "server" to the next hop IP address 7.100.100.101 (the IP address of the SSL proxy VLAN 7 on SSL-module-1).

  ```
  cat6k-router-1(config)# route-map server permit 10
  cat6k-route(config-route-map)# match ip address server
  cat6k-route(config-route-map)# set ip next-hop 7.100.100.101
  cat6k-route(config-route-map)# exit
  ```

The following example shows how to configure the routed interface and assign the route map:

```
cat6k-router-1(config)# interface Vlan7
cat6k-router-1(config-if)# ip address 7.100.100.100 255.0.0.0
cat6k-router-1(config-if)# ip policy route-map client
cat6k-router-1(config-if)# exit
cat6k-router-1(config)# interface GigabitEthernet1/1
cat6k-router-1(config-if)# ip address 217.162.1.2 255.255.255.0
cat6k-router-1(config-if)# ip policy route-map server
cat6k-router-1(config-if)# exit
```

SSL Module 1 Configuration

The following examples show how to configure the SSL client proxy service. The client proxy service is configured with virtual IP address 191.162.2.8, TCP port 80. The server is configured with IP address 7.100.100.100 so that server-side traffic is sent to 7.100.100.100 for further routing without changing the server IP address. See the "SSL Client Proxy Services" section on page 3-42 for information on configuring client proxy services. See the "Server Certificate Authentication" section on page 3-48 for more information on authenticating server certificates.

```
ssl-module1# config t
ssl-module1(config)# ssl-proxy service encrypt-clear-text client
ssl-module1(config-ssl-proxy)# virtual ipaddr 191.162.2.8 protocol tcp port 80 secondary
ssl-module1(config-ssl-proxy)# server ipaddr 7.100.100.100 protocol tcp port 443
ssl-module1(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert2048
ssl-module1(config-ssl-proxy)# no nat server
ssl-module1(config-ssl-proxy)# trusted-ca root-ca
ssl-module1(config-ssl-proxy)# authenticate verify all
ssl-module1(config-ssl-proxy)# inservice
ssl-module1(config-ssl-proxy)#
ssl-module1(config-ssl-proxy)# exit
```

The following example shows how to configure the SSL proxy VLAN on the SSL Services Module:

```
ssl-module1(config)#
ssl-module1(config)# ssl-proxy vlan 7
ssl-module1(config-vlan)# ipaddr 7.100.100.101 255.0.0.0
ssl-module1(config-vlan)# gateway 7.100.100.100
ssl-module1(config-vlan)#
```

The following example shows how to import the root-ca certificate to the SSL Services Module:

```
ssl-module1(config-vlan)# crypto ca trustpoint root-ca
ssl-module1(ca-trustpoint)# enroll ter
ssl-module1(ca-trustpoint)# exit
ssl-module1(config)# crypto ca auth root-ca

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIC1zCCAoGgAwIBAgIQadUxzU/i97hDmZRYJ1bBcDANBgkqhkiG9w0BAQUFADB1
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKY2FsaWZvcm5pYTERMA8GA1UEBxMIc2Fu
IGpvc2UxDjAMBgNVBAoTBWNpc2NvMQwwCgYDVQQLEwNoc3MxIDAeBgNVBAMTF3Np
bXBzb24tZGV2dGVzdC1yb290LUNBMB4XDTAzMTExMTIxNDgwMloXDTEzMTExMTIx
NTczOVowdTELMAkGA1UEBhMCVVMxEzARBgNVBAgTCmNhbGlmb3JuaWExETAPBgNV
BAcTCHNhbiBqb3NlMQ4wDAYDVQQKEwVjaXNjbzEMMAoGA1UECxMDaHNzMSAwHgYD
VQQDExdzaW1wc29uLWRldnRlc3Qtcm9vdC1DQTBcMA0GCSqGSIb3DQEBAQUAA0sA
MEgCQQCWEibAnUlVqQNUn0Wb94qnHi8FKjmVhibLHGRl6J+V7gHgzmF2MTz5WP5l
VQ2/1NVu0HjUORRdeCm1/raKJ/7ZAgMBAAGjgewwgekwCwYDVR0PBAQDAgHGMA8G
A1UdEwEB/wQFMAMBAf8wHQYDVR0OBBYEFCYGLUBTKNd9EgUonHnoSvbHg0axMIGX
BgNVHR8EgY8wgYwwQ6BBoD+GPWh0dHA6Ly9jaXNjby1sOGo2b2hwbnIvQ2VydEVu
cm9sbC9zaW1wc29uLWRldnRlc3Qtcm9vdC1DQS5jcmwwRaBDoEGGP2ZpbGU6Ly9c
XGNpc2NvLWw4ajZvaHBuclxDZXJ0RW5yb2xsXHNpbXBzb24tZGV2dGVzdC1yb290
LUNBLmNybDAQBgkrBgEEAYI3FQEEAwIBADANBgkqhkiG9w0BAQUFAANBACBqe1wy
YjalelGZqLVu4bDVMFo6ELCV2AMBgi41K3ix+Z/03PJd7ct2BIAF4lktv9pCe6IO
EoBcmZteA+TQcKg=
-----END CERTIFICATE-----
quit
Certificate has the following attributes:
Fingerprint:AC6FC55E CC29E891 0DC3FAAA B4747C10
% Do you accept this certificate? [yes/no]:yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

The following example shows how to configure a certificate authority pool. See the "Client Certificate Authentication" section on page 3-45 for information on configuring certificate authority pools.

```
ssl-module1(config)# ssl-proxy pool ca root-ca
ssl-module1(config-ca-pool)# ca trustpoint root
ssl-module1(config-ca-pool)# !
```

The following example shows to disable certificate revocation list (CRL) checking by entering the **crl optional** command for the trustpoint. See the "Certificate Revocation List" section on page 3-51 for information on configuring certificate revocation list options.

```
ssl-module1(config-ca-pool)# crypto ca trustpoint cert1024
ssl-module1(ca-trustpoint)# crl optional
ssl-module1(ca-trustpoint)# exit
ssl-module1(config)# exit
ssl-module1#
```

Site 2 Configuration

Site 2 in Figure A-6 shows the SSL Services Module (SSL module 2) installed in slot 3 in Catalyst 6500 switch 2.

The following example shows how to add VLAN 190 between SSL Services Module and the supervisor engine:

```
cat6k-router-2# show mod 3
Mod Ports Card Type                              Model              Serial No.
--- ----- -------------------------------------- ------------------ -----------
  3     1  SSL Module                             WS-SVC-SSL-1       SAD0722010N

Mod MAC addresses                          Hw     Fw           Sw           Status
--- ------------------------------------ ------ ------------ ------------ -------
  3  0002.fcbe.91f0 to 0002.fcbe.91f7   2.0    7.2(1)       2.1(1)       Ok

Mod Online Diag Status
--- -------------------
  3 Bypass

cat6k-router-2# config t
cat6k-router-2(config)# ssl-proxy module 3 allowed-vlan 190
cat6k-router-2(config)# exit
```

The following example shows how to configure the VLAN, configure the server port as a switchport, and assign the switchport to the access VLAN:

```
cat6k-router-2# config t
cat6k-router-2(config-vlan)# vlan 190
cat6k-router-2(config)# exit
cat6k-router-2# config t
cat6k-router-2(config)# interface GigabitEthernet10/2
cat6k-router-2(config-if)# switchport
cat6k-router-2(config-if)# switchport access vlan 190
cat6k-router-2(config-if)# switchport mode access
cat6k-router-2(config-if)# spanning-tree portfast
cat6k-router-2(config-if)# exit
cat6k-router-2(config)#
```

The following examples show how to configure the access lists:

- Access list "client" is used to match traffic going to host 191.162.2.8 with destination TCP port 443 (the standard SSL port number).

```
cat6k-router-2(config)# ip access-list extended client
cat6k-router(config-ext-nacl)# permit tcp any host 191.162.2.8 eq 443
cat6k-router(config-ext-nacl)# exit
cat6k-router-2(config)#
```

- Access list "server" is used to match traffic from server 191.162.2.8 with source port 80 (HTTP traffic).

```
cat6k-router-2(config)# ip access-list extended server
cat6k-router(config-ext-nacl)# permit tcp host 191.162.2.8 eq 80 any
cat6k-router(config-ext-nacl)# exit
cat6k-router-2(config)#
```

The following examples show how to configure route maps to redirect traffic to the SSL Services Module for encryption and decryption:

- Route map "client" redirects the traffic that matches access-list "client" to the next hop IP address 191.162.2.11 (the IP address of SSL proxy VLAN 190 on SSL-module-2). This configuration redirects encrypted HTTP traffic to the SSL Services Module for decryption.

```
cat6k-router-2(config)# route-map client permit 10
cat6k-route(config-route-map)# match ip address client
cat6k-route(config-route-map)# set ip next-hop 191.162.2.11
cat6k-route(config-route-map)# exit
cat6k-router-2(config)#
```

- Route map "server" redirects the traffic that matches access-list "server" to the next hop IP address 191.162.2.11 (the IP address of the SSL proxy VLAN 190 on SSL-module-2). This configuration redirects clear text HTTP traffic to the SSL Services Module for encryption.

```
cat6k-router-2(config)# route-map server permit 10
cat6k-route(config-route-map)# match ip address server
cat6k-route(config-route-map)# set ip next-hop 191.162.2.11
cat6k-route(config-route-map)# exit
cat6k-router-2(config)#
```

The following example shows how to configure the routed-interface and assign the IP policy route maps:

```
cat6k-router-2(config)# interface GigabitEthernet1/1
cat6k-router-2(config-if)# ip address 217.162.1.1 255.255.255.0
cat6k-router-2(config-if)# ip policy route-map client
cat6k-router-2(config-if)# exit
cat6k-router-2(config)#
cat6k-router-2(config-if)# interface Vlan190
cat6k-router-2(config-if)# ip address 191.162.2.10 255.0.0.0
cat6k-router-2(config-if)# ip policy route-map server
cat6k-router-2(config-if)# exit
cat6k-router-2(config)# exit
```

SSL Module 2 Configuration

The following example shows how to configure the SSL server proxy to decrypt the encrypted HTTP traffic into clear text HTTP traffic:

```
ssl-module2# config t
ssl-module2(config)# ssl-proxy service decrypt-ssl-traffic
ssl-module2(config-ssl-proxy)# virtual ipaddr 191.162.2.8 protocol tcp port 443 secondary
ssl-module2(config-ssl-proxy)# server ipaddr 191.162.2.10 protocol tcp port 80
ssl-module2(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert1024
ssl-module2(config-ssl-proxy)# no nat server
ssl-module2(config-ssl-proxy)# trusted-ca root-ca
ssl-module2(config-ssl-proxy)# authenticate verify all
ssl-module2(config-ssl-proxy)# inservice
ssl-module2(config-ssl-proxy)# exit
ssl-module2(config)#
```

This example shows how to configure SSL proxy VLAN:

```
ssl-module2(config)# ssl-proxy vlan 190
ssl-module2(config-vlan)# ipaddr 191.162.2.11 255.255.0.0
ssl-module2(config-vlan)# gateway 191.162.2.10
ssl-module2(config-vlan)# exit
ssl-module2(config)#
```

The following example shows how to import the root-ca certificate to the SSL Services Module:

```
ssl-module2(config)# crypto ca trustpoint root-ca
ssl-module2(ca-trustpoint)# crl optional
ssl-module2(ca-trustpoint)# enrollment terminal
ssl-module2(ca-trustpoint)# exit
ssl-module2(config)# crypto ca authenticate root-ca

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

-----BEGIN CERTIFICATE-----
MIIC1zCCAoGgAwIBAgIQadUxzU/i97hDmZRYJ1bBcDANBgkqhkiG9w0BAQUFADB1
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKY2FsaWZvcm5pYTERMA8GA1UEBxMIc2Fu
IGpvc2UxDjAMBgNVBAoTBWNpc2NvMQwwCgYDVQQLEwNoc3MxIDAeBgNVBAMTF3Np
bXBzb24tZGV2dGVzdC1yb290LUNBMB4XDTAzMTExMTIxNDgwMloXDTEzMTExMTIx
NTczOVowdTELMAkGA1UEBhMCVVMxEzARBgNVBAgTCmNhbGlmb3JuaWExETAPBgNV
BAcTCHNhbiBqb3NlMQ4wDAYDVQQKEwVjaXNjbzEMMAoGA1UECxMDaHNzMSAwHgYD
VQQDExdzaW1wc29uLWRldnRlc3Qtcm9vdC1DQTBcMA0GCSqGSIb3DQEBAQUAA0sA
MEgCQQCWEibAnUlVqQNUn0Wb94qnHi8FKjmVhibLHGRl6J+V7gHgzmF2MTz5WP5l
VQ2/1NVu0HjUORRdeCm1/raKJ/7ZAgMBAAGjgewwgekwCwYDVR0PBAQDAgHGMA8G
A1UdEwEB/wQFMAMBAf8wHQYDVR0OBBYEFCYGLUBTKNd9EgUonHnoSvbHg0axMIGX
BgNVHR8EgY8wgYwwQ6BBoD+GPWh0dHA6Ly9jaXNjby1sOGo2b2hwbnIvQ2VydEVu
cm9sbC9zaW1wc29uLWRldnRlc3Qtcm9vdC1DQS5jcmwwRaBDoEGGP2ZpbGU6Ly9c
XGNpc2NvLWw4ajZvaHBuclxDZXJ0RW5yb2xsXHNpbXBzb24tZGV2dGVzdC1yb290
LUNBLmNybDAQBgkrBgEEAYI3FQEEAwIBADANBgkqhkiG9w0BAQUFAANBACBqe1wy
YjalelGZqLVu4bDVMFo6ELCV2AMBgi41K3ix+Z/03PJd7ct2BIAF4lktv9pCe6IO
EoBcmZteA+TQcKg=
-----END CERTIFICATE-----
quit
Certificate has the following attributes:
Fingerprint:AC6FC55E CC29E891 0DC3FAAA B4747C10
% Do you accept this certificate? [yes/no]:yes
Trustpoint CA certificate accepted.
% Certificate successfully imported

ssl-module2(config)#
```

The following example shows how to configure a certificate authority pool. See the "Client Certificate Authentication" section on page 3-45 for information on configuring certificate authority pools. The example also show to disable certificate revocation list (CRL) checking by entering the **crl optional** command for the trustpoint. See the "Certificate Revocation List" section on page 3-51 for information on configuring certificate revocation list options.

```
ssl-module2(config)#
ssl-module2(config)# ssl-proxy pool ca root-ca
ssl-module2(config-ca-pool)# ca trustpoint root-ca
ssl-module2(config-ca-pool)# exit
ssl-module2(config)# crypto ca trustpoint cert1024
ssl-module2(ca-trustpoint)# crl optional
ssl-module2(ca-trustpoint)# exit
ssl-module2(config)#
```

The following example show how to display statistics when connections are active:

- SSL module 1

```
ssl-module1# show ssl-proxy con
Connections for TCP module 1
Local Address         Remote Address        VLAN Conid  Send-Q Recv-Q State
--------------------- --------------------- ---- ------ ------ ------ ------
191.162.2.8:80        7.100.100.1:34472     7    9      0      0      ESTAB
7.100.100.1:34472     191.162.2.8:443       7    196617 0      0      ESTAB
```

- SSL module 12

```
ssl-module2# show ssl-proxy con
Connections for TCP module 1
Local Address         Remote Address        VLAN Conid  Send-Q Recv-Q State
--------------------- --------------------- ---- ------ ------ ------ ------
191.162.2.8:443       7.100.100.1:34472     190  9      0      0      ESTAB
7.100.100.1:34472     191.162.2.8:80        190  196617 0      0      ESTAB
```

Certificate Security Attribute-Based Access Control Examples

The Certificate Security Attribute-Based Access Control feature adds fields to the certificate that allow specifying an access control list (ACL), to create a certificate-based ACL.

For information on configuring certificate security attribute-based access control, refer to *Certificate Security Attribute-Based Access Control* at this URL:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t15/ftcrtacl.htm

In the following example, SSL connections for the SSL proxy service "ssl-offload" are successful only if the subject-name of the client certificate contains the domain name **.cisco.com**:

```
ssl-proxy(config)# ssl-proxy service ssl-offload
ssl-proxy(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 80
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-proxy(config-ssl-proxy)# nat client client-nat
ssl-proxy(config-ssl-proxy)# trusted-ca root-ca
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# exit
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy pool ca root-ca
ssl-proxy(config-ca-pool)# ca trustpoint root
ssl-proxy(config-ca-pool)# exit
ssl-proxy(config)#
```

```
ssl-proxy(config)# crypto ca trustpoint root
ssl-proxy(ca-trustpoint)# enrollment mode ra
ssl-proxy(ca-trustpoint)# enrollment terminal
ssl-proxy(ca-trustpoint)# crl optional
ssl-proxy(ca-trustpoint)# match certificate acl
ssl-proxy(ca-trustpoint)# exit
ssl-proxy(config)#
ssl-proxy(config)# crypto ca certificate map acl 10
ssl-proxy(ca-certificate-map)# subject-name co .cisco.com
ssl-proxy(ca-certificate-map)# exit
```

In the following example, certificate ACLs are configured so that SSL connections for the proxy service "ssl-offload" are successful for the following conditions:

- the subject-name of the client certificate contains **ste3-server.cisco.com** or **ste2-server.cisco.com**
- the valid-start of the client certificate is greater than or equal to 30th Jul 2003
- the expiration date of the client certificate is less than 1st Jan 2007
- the issuer-name of the client certificate contains the string "certificate manager"

```
ssl-proxy(config)# ssl-proxy service ssl-offload
ssl-proxy(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 80
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-proxy(config-ssl-proxy)# nat client client-nat
ssl-proxy(config-ssl-proxy)# trusted-ca root-ca
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# exit
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy pool ca root-ca
ssl-proxy(config-ca-pool)# ca trustpoint root
ssl-proxy(config-ca-pool)# exit
ssl-proxy(config)#
ssl-proxy(config)# crypto ca trustpoint root
ssl-proxy(ca-trustpoint)# enrollment mode ra
ssl-proxy(ca-trustpoint)# enrollment terminal
ssl-proxy(ca-trustpoint)# crl optional
ssl-proxy(ca-trustpoint)# match certificate acl
ssl-proxy(ca-trustpoint)# exit
ssl-proxy(config)#
ssl-proxy(config)# crypto ca certificate map acl 10
ssl-proxy(ca-certificate-map)# subject-name co ste3-server.cisco.com
ssl-proxy(ca-certificate-map)# valid-start ge Jul 30 2003 00:00:00 UTC
ssl-proxy(ca-certificate-map)# expires-on lt Jan 01 2007 00:00:00 UTC
ssl-proxy(ca-certificate-map)# issuer-name co certificate manager
ssl-proxy(ca-certificate-map)# exit
ssl-proxy(config)#
ssl-proxy(config)# crypto ca certificate map acl 20
ssl-proxy(ca-certificate-map)# subject-name co ste2-server.cisco.com
ssl-proxy(ca-certificate-map)# expires-on lt Jan 01 2007 00:00:00 UTC
ssl-proxy(ca-certificate-map)# issuer-name co certificate manager
ssl-proxy(ca-certificate-map)# valid-start ge Jul 30 2003 00:00:00 UTC
ssl-proxy(ca-certificate-map)# exit
```

In the following SSL initiation example, the server certificate is checked for the domain name in the certificate field. SSL initiation is successful only if the subject-name of the server certificate contains the domain name **.cisco.com**:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy service ssl-initiation client
ssl-proxy(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 81
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# nat client client-nat
ssl-proxy(config-ssl-proxy)# trusted-ca root
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# exit
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy pool ca root-ca
ssl-proxy(config-ca-pool)# ca trustpoint root
ssl-proxy(config-ca-pool)# exit
ssl-proxy(config)#
ssl-proxy(config)# crypto ca trustpoint root
ssl-proxy(ca-trustpoint)# enrollment mode ra
ssl-proxy(ca-trustpoint)# enrollment terminal
ssl-proxy(ca-trustpoint)# crl optional
ssl-proxy(ca-trustpoint)# match certificate acl
ssl-proxy(ca-trustpoint)# exit
ssl-proxy(config)#
ssl-proxy(config)# crypto ca certificate map acl 10
ssl-proxy(ca-certificate-map)# subject-name co .cisco.com
ssl-proxy(ca-certificate-map)# exit
ssl-proxy(config)#
```

HTTP Header Insertion Examples

The following examples show how to insert various HTTP headers and how to display header insertion statistics.

Example 1

This example shows how to insert custom headers, client IP address and TCP port number information, and a prefix string in HTTP requests sent to the server:

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy policy http-header ssl-offload
ssl-proxy(config-http-header-policy)# custom "SOFTWARE VERSION :2.1(1)"
ssl-proxy(config-http-header-policy)# custom "module :SSL MODULE - CATALYST 6500"
ssl-proxy(config-http-header-policy)# custom
type-of-proxy:server_proxy_with_1024_bit_key_size
ssl-proxy(config-http-header-policy)# client-ip-port
ssl-proxy(config-http-header-policy)# prefix SSL-OFFLOAD
ssl-proxy(config-http-header-policy)# exit
ssl-proxy(config)#
```

```
ssl-proxy(config)# ssl-proxy service ssl-offload
ssl-proxy(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 80
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-proxy(config-ssl-proxy)# nat client client-nat
ssl-proxy(config-ssl-proxy)# policy http-header ssl-offload
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# exit
ssl-proxy(config)# exit
```

Custom headers and client IP address and TCP port number information are added to every HTTP request and are prefixed by the prefix string, as shown below:

```
SSL-OFFLOAD-Client-IP:7.100.100.1
SSL-OFFLOAD-Client-Port:59008
SSL-OFFLOAD-SOFTWARE VERSION :2.1(1)
SSL-OFFLOAD-module :SSL MODULE - CATALYST 6500
SSL-OFFLOAD-type-of-proxy:server_proxy_with_1024_bit_key_size
```

This example shows how to display header insertion information:

```
ssl-proxy# show ssl-proxy stats hdr
Header Insert Statistics:
    Session Headers Inserted :0          Custom Headers Inserted :2
    Session Id's Inserted    :0          Client Cert. Inserted   :0
    Client IP/Port Inserted  :2
    No End of Hdr Detected   :0          Payload no HTTP header  :0
    Desc Alloc Failed        :0          Buffer Alloc Failed     :0
    Client Cert Errors       :0          No Service              :0
```

This example shows how to display SSL statistics:

```
ssl-proxy# show ssl-proxy stats ssl
SSL Statistics:
    conns attempted     :2             conns completed     :2
    conns in handshake  :0             conns in data       :0
    renegs attempted    :0             conns in reneg      :0
    active sessions     :0             max handshake conns :1
    rand bufs allocated :0             cached rand buf miss:0
    current device q len:0             max device q len    :2
    sslv2 forwards      :0             cert reqs processed :0
    fatal alerts rcvd   :0             fatal alerts sent   :0
    stale packet drops  :0             service_id discards :0
    session reuses      :0

    SSL3 Statistics:
      full handshakes    :0             resumed handshakes :0
      handshake failures :0             data failures      :0
      bad macs received  :0             pad errors         :0
      conns established with cipher rsa-with-rc4-128-md5        :0
      conns established with cipher rsa-with-rc4-128-sha        :0
      conns established with cipher rsa-with-des-cbc-sha        :0
      conns established with cipher rsa-with-3des-ede-cbc-sha   :0

    TLS1 Statistics:
      full handshakes    :1             resumed handshakes :1
      handshake failures :0             data failures      :0
      bad macs received  :0             pad errors         :0
      conns established with cipher rsa-with-rc4-128-md5        :0
      conns established with cipher rsa-with-rc4-128-sha        :2
      conns established with cipher rsa-with-des-cbc-sha        :0
      conns established with cipher rsa-with-3des-ede-cbc-sha   :0
```

Example 2

This example shows how to insert session headers and a prefix string. The full session headers are added to the HTTP request when the full SSL handshake occurs. However, only the session ID is inserted when the session resumes.

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy policy http-header ssl-offload
ssl-proxy(config-http-header-policy)# session
ssl-proxy(config-http-header-policy)# prefix SSL-OFFLOAD
ssl-proxy(config-http-header-policy)# exit
ssl-proxy(config)#
ssl-proxy(config)# ssl-proxy service ssl-offload
ssl-proxy(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 80
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-proxy(config-ssl-proxy)# nat client client-nat
ssl-proxy(config-ssl-proxy)# policy http-header ssl-offload
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# exit
ssl-proxy(config)# exit
```

For the full SSL handshake, the session headers, prefixed by the prefix string, are added to the HTTP request as shown below:

```
SSL-OFFLOAD-Session-Id:33:FF:2C:2D:25:15:3C:50:56:AB:FA:5A:81:0A:EC:E9:00:00:0A:03:00:60:
2F:30:9C:2F:CD:56:2B:91:F2:FF
SSL-OFFLOAD-Session-Cipher-Name:RC4-SHA
SSL-OFFLOAD-Session-Cipher-Key-Size:128
SSL-OFFLOAD-Session-Cipher-Use-Size:128
```

When the session resumes, only the session ID is inserted:

```
SSL-OFFLOAD-Session-Id:33:FF:2C:2D:25:15:3C:50:56:AB:FA:5A:81:0A:EC:E9:00:00:0A:03:00:60:
2F:30:9C:2F:CD:56:2B:91:F2:FF
```

This example shows how to display header insertion information:

```
ssl-proxy# show ssl-proxy stats hdr
Header Insert Statistics:
    Session Headers Inserted :1          Custom Headers Inserted :0
    Session Id's Inserted    :2          Client Cert. Inserted   :0
    Client IP/Port Inserted  :0
    No End of Hdr Detected   :0          Payload no HTTP header  :0
    Desc Alloc Failed        :0          Buffer Alloc Failed     :0
    Client Cert Errors       :0          No Service              :0
```

This example shows how to display SSL statistics:

```
ssl-proxy# show ssl-proxy stats ssl
SSL Statistics:
    conns attempted     :2             conns completed     :2
    conns in handshake  :0             conns in data       :0
    renegs attempted    :0             conns in reneg      :0
    active sessions     :0             max handshake conns :1
    rand bufs allocated :0             cached rand buf miss:0
    current device q len:0             max device q len    :2
    sslv2 forwards      :0             cert reqs processed :0
    fatal alerts rcvd   :0             fatal alerts sent   :0
    stale packet drops  :0             service_id discards :0
    session reuses      :0
```

```
SSL3 Statistics:
  full handshakes    :0             resumed handshakes :0
  handshake failures :0             data failures      :0
  bad macs received  :0             pad errors         :0
  conns established with cipher rsa-with-rc4-128-md5        :0
  conns established with cipher rsa-with-rc4-128-sha        :0
  conns established with cipher rsa-with-des-cbc-sha        :0
  conns established with cipher rsa-with-3des-ede-cbc-sha   :0

TLS1 Statistics:
  full handshakes    :1             resumed handshakes :1
  handshake failures :0             data failures      :0
  bad macs received  :0             pad errors         :0
  conns established with cipher rsa-with-rc4-128-md5        :0
  conns established with cipher rsa-with-rc4-128-sha        :2
  conns established with cipher rsa-with-des-cbc-sha        :0
  conns established with cipher rsa-with-3des-ede-cbc-sha   :0
```

Example 3

This example shows how to insert custom headers, decoded client certificate fields, and the IP address and destination TCP port number of the client-side connection, prefixed by the prefix string. The complete decoded client certificate fields are inserted for the full SSL handshake. However, only session ID is inserted when the SSL session resumes.

```
ssl-proxy# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-proxy(config)# ssl-proxy policy http-header ssl-offload
ssl-proxy(config-http-header-policy)# custom "SOFTWARE VERSION :2.1(1)"
ssl-proxy(config-http-header-policy)# custom "module :SSL MODULE - CATALYST 6500"
ssl-proxy(config-http-header-policy)# custom
type-of-proxy:server_proxy_with_1024_bit_key_size
ssl-proxy(config-http-header-policy)# client-cert
ssl-proxy(config-http-header-policy)# client-ip-port
ssl-proxy(config-http-header-policy)# prefix SSL-OFFLOAD
ssl-proxy(config-http-header-policy)# exit
ssl-proxy(config)# ssl-proxy service ssl-offload
ssl-proxy(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443
ssl-proxy(config-ssl-proxy)# server ipaddr 191.162.2.8 protocol tcp port 80
ssl-proxy(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-proxy(config-ssl-proxy)# nat client client-nat
ssl-proxy(config-ssl-proxy)# policy http-header ssl-offload
ssl-proxy(config-ssl-proxy)# trusted-ca root-ca
ssl-proxy(config-ssl-proxy)# authenticate verify all
ssl-proxy(config-ssl-proxy)# inservice
ssl-proxy(config-ssl-proxy)# exit
ssl-proxy(config)# exit
```

For the full SSL handshake, the custom headers, decoded client certificate fields, the IP address and destination TCP port number of the client-side connection, prefixed by the prefix string, are added to the HTTP request, as shown below:

```
SSL-OFFLOAD-Client-IP:7.100.100.1
SSL-OFFLOAD-Client-Port:59011
SSL-OFFLOAD-Session-Id:0F:61:9C:F2:E5:98:70:9D:1B:C1:EA:1D:38:F5:A1:2B:00:00:0E:03:00:60:
2F:30:9C:2F:1D:7D:5A:82:30:F6
SSL-OFFLOAD-SOFTWARE VERSION :2.1(1)
SSL-OFFLOAD-module :SSL MODULE - CATALYST 6500
SSL-OFFLOAD-type-of-proxy:server_proxy_with_1024_bit_key_size
SSL-OFFLOAD-ClientCert-Valid:1
SSL-OFFLOAD-ClientCert-Error:none
```

```
SSL-OFFLOAD-ClientCert-Fingerprint:1B:11:0F:E8:20:3F:6C:23:12:9C:76:C0:C1:C2:CC:85
SSL-OFFLOAD-ClientCert-Subject-CN:a
SSL-OFFLOAD-ClientCert-Issuer-CN:Certificate Manager
SSL-OFFLOAD-ClientCert-Certificate-Version:3
SSL-OFFLOAD-ClientCert-Serial-Number:0F:E5
SSL-OFFLOAD-ClientCert-Data-Signature-Algorithm:sha1WithRSAEncryption
SSL-OFFLOAD-ClientCert-Subject:OID.1.2.840.113549.1.9.2 = ste2-server.cisco.com +
OID.2.5.4.5 = B0FFF22E, CN = a, O = Cisco
SSL-OFFLOAD-ClientCert-Issuer:CN = Certificate Manager, OU = HSS, O = Cisco, L = San Jose,
ST = California, C = US
SSL-OFFLOAD-ClientCert-Not-Before:22:29:26 UTC Jul 30 2003
SSL-OFFLOAD-ClientCert-Not-After:07:00:00 UTC Apr 27 2006
SSL-OFFLOAD-ClientCert-Public-Key-Algorithm:rsaEncryption
SSL-OFFLOAD-ClientCert-RSA-Public-Key-Size:1024 bit
SSL-OFFLOAD-ClientCert-RSA-Modulus-Size:1024 bit
SSL-OFFLOAD-ClientCert-RSA-Modulus:B3:32:3C:5E:C9:D1:CC:76:FF:81:F6:F7:97:58:91:4D:B2:0E:
C1:3A:7B:62:63:BD:5D:F6:5F:68:F0:7D:AC:C6:72:F5:72:46:7E:FD:38:D3:A2:E1:03:8B:EC:F7:C9:9A:
80:C7:37:DA:F3:BE:1F:F4:5B:59:BD:52:72:94:EE:46:F5:29:A4:B3:9B:2E:4C:69:D0:11:59:F7:68:3A:
D9:6E:ED:6D:54:4E:B5:A7:89:B9:45:9E:66:0B:90:0B:B1:BD:F4:C8:15:12:CD:85:13:B2:0B:FE:7E:8D:
F0:D7:4A:98:BB:08:88:6E:CC:49:60:37:22:74:4D:73:1E:96:58:91
SSL-OFFLOAD-ClientCert-RSA-Exponent:00:01:00:01
SSL-OFFLOAD-ClientCert-X509v3-Authority-Key-Identifier:keyid=EE:EF:5B:BD:4D:CD:F5:6B:60:
9D:CF:46:C2:EA:25:7B:22:A5:08:00
SSL-OFFLOAD-ClientCert-X509v3-Basic-Constraints:
SSL-OFFLOAD-ClientCert-Signature-Algorithm:sha1WithRSAEncryption
SSL-OFFLOAD-ClientCert-Signature:87:09:C1:F8:86:C1:15:C5:57:18:8E:B3:0D:62:E1:0F:6F:D4:9D:
75:DA:5D:53:E2:C6:0B:73:99:61:BE:B0:F6:19:83:F2:E5:48:1B:D2:6C:92:83:66:B3:63:A6:58:B4:5C:
0E:5D:1B:60:F9:86:AF:B3:93:07:77:16:74:4B:C5
```

This example shows how to display header insertion information:

```
ssl-proxy# show ssl-proxy stats hdr
Header Insert Statistics:
    Session Headers Inserted :0          Custom Headers Inserted :1
    Session Id's Inserted    :1          Client Cert. Inserted   :1
    Client IP/Port Inserted  :1
    No End of Hdr Detected   :0          Payload no HTTP header  :0
    Desc Alloc Failed        :0          Buffer Alloc Failed     :0
    Client Cert Errors       :0          No Service              :0
```

This example shows how to display SSL statistics:

```
ssl-proxy# show ssl-proxy stats ssl
SSL Statistics:
    conns attempted     :1             conns completed     :1
    conns in handshake  :0             conns in data       :0
    renegs attempted    :0             conns in reneg      :0
    active sessions     :0             max handshake conns :1
    rand bufs allocated :0             cached rand buf miss:0
    current device q len:0             max device q len    :2
    sslv2 forwards      :0             cert reqs processed :1
    fatal alerts rcvd   :0             fatal alerts sent   :0
    stale packet drops  :0             service_id discards :0
    session reuses      :0

    SSL3 Statistics:
      full handshakes    :0             resumed handshakes :0
      handshake failures :0             data failures      :0
      bad macs received  :0             pad errors         :0
      conns established with cipher rsa-with-rc4-128-md5        :0
      conns established with cipher rsa-with-rc4-128-sha        :0
      conns established with cipher rsa-with-des-cbc-sha        :0
      conns established with cipher rsa-with-3des-ede-cbc-sha   :0
```

```
TLS1 Statistics:
  full handshakes    :1             resumed handshakes :0
  handshake failures :0             data failures      :0
  bad macs received  :0             pad errors         :0
  conns established with cipher rsa-with-rc4-128-md5        :0
  conns established with cipher rsa-with-rc4-128-sha        :0
  conns established with cipher rsa-with-des-cbc-sha        :0
  conns established with cipher rsa-with-3des-ede-cbc-sha   :1
```

URL Rewrite Examples

The following examples show how to configure URL rewrite depending on the desired outcome and assume the following proxy configuration:

```
ssl-proxy service frontend
 virtual ipaddr 35.200.200.101 protocol tcp port 443
 server ipaddr 35.200.200.14 protocol tcp port 80
 certificate rsa general-purpose trustpoint TP-1024-pkcs12
 policy url-rewrite test-url-rewrite
 inservice
!
```

Example 1

The following example shows how to configure a protocol rewrite (for example, HTTP to HTTPS) when the clear text port is the standard HTTP port 80. In this example, when the server sends the relocation string as http://ssl-136.cisco.com/index2.html, the SSL Services Module rewrites the string as https://ssl-136.cisco.com/index2.html.

To configure a protocol rewrite (HTTP to HTTPS), specify any of the following URL rewrite rules:

-
```
ssl-proxy policy url-rewrite test-url-rewrite
 url ssl-136.cisco.com
!
```
-
```
ssl-proxy policy url-rewrite test-url-rewrite
 url ssl*
!
```
-
```
Ssl-proxy policy url-rewrite test-url-rewrite
 url *com
!
```

Example 2

The following example shows how to configure a protocol rewrite (for example, HTTP to HTTPS) when the clearport is a non-standard HTTP port. In this example, when the server sends the relocation string as http://ssl-136.cisco.com:100/index2.html, the SSL Services Module rewrites the string as https://ssl-136.cisco.com/index2.html.

To configure a protocol rewrite (HTTP to HTTPS) with a non-standard clear text port, specify any of the following URL rewrite rules:

- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url ssl-136.cisco.com clearport 100
  !
  ```
- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url ssl* clearport 100
  !
  ```
- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url *com clearport 100
  !
  ```

Example 3

The following example shows how to configure a protocol rewrite and SSL port rewrite when the clear text port is the standard HTTP port 80. In this example, when the server sends the relocation string as http://ssl-136.cisco.com/index2.html, the SSL Services Module rewrites the string as https://ssl-136.cisco.com:445/index2.html.

To configure a protocol rewrite (HTTP to HTTPS) with a non-standard SSL text port, specify any of the following URL rewrite rules:

- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url ssl-136.cisco.com sslport 445
  !
  ```
- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url ssl* sslport 445
  !
  ```
- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url *com sslport 445
  !
  ```

Example 4

The following example shows how to configure a protocol rewrite and SSL port rewrite when the clear text port is non-standard. In this example, when the server sends the relocation string as http://ssl-136.cisco.com:100/index2.html, the SSL Services Module rewrites the string as https://ssl-136.cisco.com:445/index2.html.

To configure a protocol rewrite and SSL port rewrite with a non-standard clear text port, specify any of the following URL rewrite rules:

- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url ssl-136.cisco.com clearport 100 sslport 445
  !
  ```
- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url ssl* clearport 100 sslport 445
  !
  ```
- ```
  ssl-proxy policy url-rewrite test-url-rewrite
   url *com clearport 100 sslport 445
  !
  ```
  ```

The following example displays the above URL rewrite policy:

```
ssl-proxy# show ssl-proxy policy url-rewrite test-url-rewrite
Rule URL                                      Clearport SSLport
  1 *com                                        100        445
SSL proxy services using this policy:
          frontend
Usage count of this policy:1

ssl-proxy#
```

HSRP Examples

In systems with an SSL Services Module and a Content Switching Module (CSM), the failover functionality on the CSM provides stateless redundancy on the SSL module. When the SSL module is used in a standalone configuration (using policy-based routing), you can configure HSRP to provide redundancy.

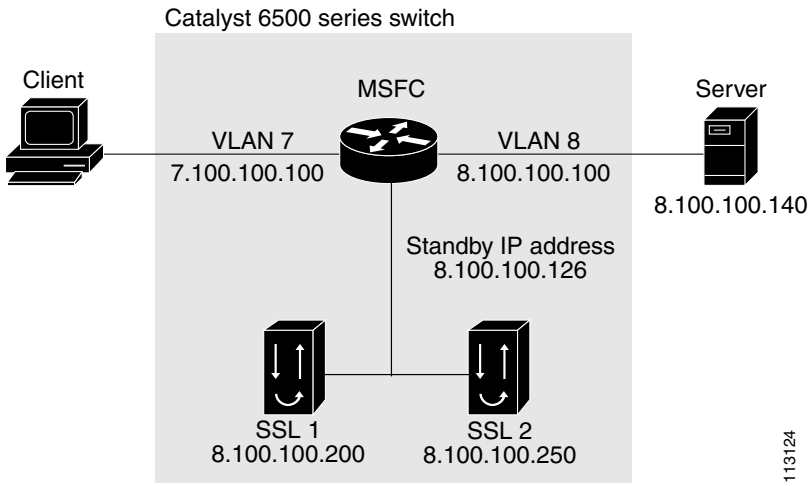
See the "Configuring Redundancy" section on page 4-12 for more information on configuring redundancy using HSRP.

Standalone Redundancy Example

In Figure A-7, both SSL Services Modules have the same proxy service configured, specifying the **secondary** keyword for the virtual IP address and the same HSRP configuration. Both modules are configured with standby IP address 8.100.100.126. SSL 1 is the active module and accepts SSL connections. SSL 2 is the backup module and does not accept SSL connections until SSL 1 goes offline.

Policy-based routing is configured on the MSFC so that any TCP traffic destined to 8.100.100.126:443 is redirected to the next-hop IP address 8.100.100.126.

Figure A-7 Standalone Redundancy



Supervisor Engine Configuration

This example shows how to configure the route maps and access lists:

```
Router# config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# route-map client permit 10
Router(config-route-map)# set ip next-hop 8.100.100.126
Router(config-route-map)# match ip address client
Router(config-route-map)# exit
Router(config)# route-map server permit 10
Router(config-route-map)# match ip address server
Router(config-route-map)# set ip next-hop 8.100.100.126
Router(config-route-map)# exit
Router(config)#
Router(config)# ip access-list extended client
Router(config-ext-nacl)# permit tcp any host 8.100.100.126 eq 443
Router(config-ext-nacl)# exit
Router(config)#
Router(config)# ip access-list extended server
Router(config-ext-nacl)# permit tcp host 8.100.100.140 eq www any
Router(config-ext-nacl)# exit
Router(config)#
Router(config)# interface Vlan7
Router(config-if)# ip address 7.100.100.100 255.0.0.0
Router(config-if)# ip policy route-map client
Router(config-if)# exit
Router(config)#
Router(config)# interface Vlan8
Router(config-if)# ip address 8.100.100.100 255.0.0.0
Router(config-if)# ip policy route-map server
Router(config-if)# exit
Router(config)# exit
Router#
```

SSL 1 Configuration

This example shows how to configure the proxy service and the VLAN on SSL 1:

```
ssl-mod-1# config t
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-mod-1(config)# ssl-proxy service ssl-offload
ssl-mod-1(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443 secondary
ssl-mod-1(config-ssl-proxy)# server ipaddr 8.100.100.140 protocol tcp port 80
ssl-mod-1(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-mod-1(config-ssl-proxy)# inservice
ssl-mod-1(config-ssl-proxy)# exit
ssl-mod-1(config)#
ssl-mod-1(config)# ssl-proxy vlan 8
ssl-mod-1(config-vlan)# ipaddr 8.100.100.200 255.0.0.0
ssl-mod-1(config-vlan)# gateway 8.100.100.100
ssl-mod-1(config-vlan)# route 191.0.0.0 255.0.0.0 gateway 8.100.100.100
ssl-mod-1(config-vlan)# standby ip 8.100.100.126
ssl-mod-1(config-vlan)# standby timers 1 3
ssl-mod-1(config-vlan)# standby priority 90
ssl-mod-1(config-vlan)# exit
ssl-mod-1(config)# exit
ssl-mod-1#
```

SSL 2 Configuration

This example shows how to configure the proxy service and the VLAN on SSL 2:

```
ssl-mod-2# config t
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-mod-2(config)# ssl-proxy service ssl-offload
ssl-mod-2(config-ssl-proxy)# virtual ipaddr 8.100.100.126 protocol tcp port 443 secondary
ssl-mod-2(config-ssl-proxy)# server ipaddr 8.100.100.140 protocol tcp port 80
ssl-mod-2(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-mod-2(config-ssl-proxy)# inservice
ssl-mod-2(config-ssl-proxy)# exit
ssl-mod-2(config)#
ssl-mod-2(config)# ssl-proxy vlan 8
ssl-mod-2(config-vlan)# ipaddr 8.100.100.250 255.0.0.0
ssl-mod-2(config-vlan)# gateway 8.100.100.100
ssl-mod-2(config-vlan)# standby ip 8.100.100.126
ssl-mod-2(config-vlan)# standby timers 1 3
ssl-mod-2(config-vlan)# standby priority 110
ssl-mod-2(config-vlan)# exit
ssl-mod-2(config)# exit
ssl-mod-2#
```

Load Balancing Example

In Figure A-8, each SSL Services Module is configured with more than one proxy service. Each SSL Services Module has a different HSRP group configured.

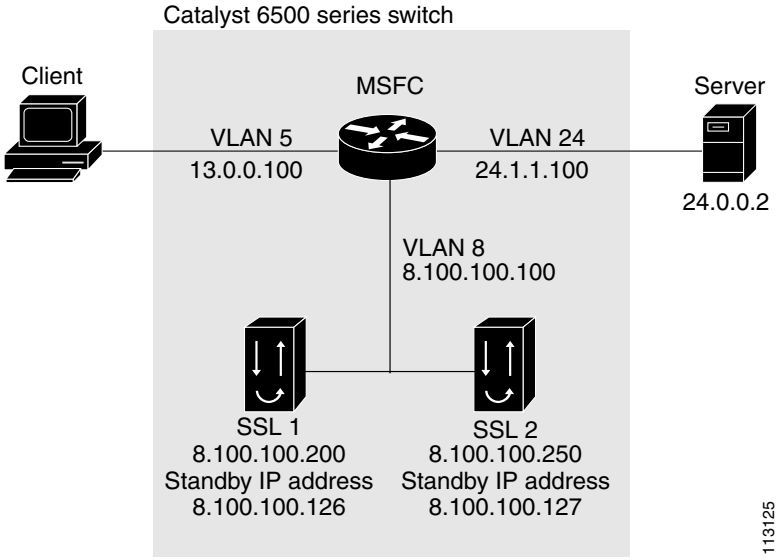
On the MSFC, configure policy-based routing so that traffic to the different proxy services is load balanced between the two SSL Services Modules.

On the SSL Services Modules, configure the **standby** *group_number* **preempt delay** *delay* command for the following reasons:

- When a module goes offline and comes back online, half of the traffic is switched back to the new (online) module for efficient load balancing.
- The new (online) module does not become immediately active, giving sufficient time for the proxy services to come up.

Configure client NAT for each proxy service so that when multiple proxies send traffic to the same server, the traffic from the server is sent back to the module that originated the traffic. See the "Client NAT" section on page 4-11 for information on configuring client NAT.

Figure A-8 Load Balancing



Supervisor Engine Configuration

This example shows how to configure the route maps and access lists:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip access-list extended ssl-offload
Router(config-ext-nacl)# permit tcp any host 8.100.100.110 eq 443
Router(config-ext-nacl)# exit
Router(config)#
Router(config)# ip access-list extended ssl-offload-checkout
Router(config-ext-nacl)# permit tcp any host 8.100.100.111 eq 443
Router(config-ext-nacl)#
Router(config-ext-nacl)# exit
Router(config)#
Router(config)# route-map client permit 10
Router(config-route-map)# match ip address ssl-offload
Router(config-route-map)# set ip next-hop 8.100.100.126
Router(config-route-map)#
Router(config-route-map)# exit
Router(config)#
Router(config)# route-map client permit 20
Router(config-route-map)# match ip address ssl-offload-checkout
Router(config-route-map)# set ip next-hop 8.100.100.127
Router(config-route-map)# exit
Router(config)#
Router(config)# interface Vlan5
Router(config-if)# ip address 13.0.0.100 255.0.0.0
Router(config-if)# ip policy route-map client
Router(config-if)# no shutdown
Router(config-if)# end
```

```
Router# config t
Router(config)# interface GigabitEthernet10/7
Router(config-if)# switchport
Router(config-if)# switchport access vlan 5
Router(config-if)# switchport mode access
Router(config-if)# spanning-tree portfast
Router(config-if)# no shutdown
Router(config-if)# end
Router#
Router# config t
Router(config)# interface GigabitEthernet10/11
Router(config-if)# switchport
Router(config-if)# switchport access vlan 24
Router(config-if)# switchport mode access
Router(config-if)# spanning-tree portfast
Router(config-if)# no shutdown
Router(config-if)# end
Router# config t
Router(config)# interface Vlan24
Router(config-if)# ip address 24.1.1.100 255.0.0.0
Router(config-if)# no shutdown
Router(config-if)# no ip redirects
Router(config-if)# ^Z
Router#
```

SSL 1 Configuration

This example shows how to configure the proxy services and the VLAN on SSL 1:

```
ssl-mod-1# config t
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-mod-1(config)# ssl-proxy natpool client-nat 8.100.1.1 8.100.1.8 netmask 255.0.0.0
ssl-mod-1(config)# ssl-proxy service ssl-offload
ssl-mod-1(config-ssl-proxy)# virtual ipaddr 8.100.100.110 protocol tcp port 443 secondary
ssl-mod-1(config-ssl-proxy)# server ipaddr 24.0.0.2 protocol tcp port 80
ssl-mod-1(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-mod-1(config-ssl-proxy)# nat client client-nat
ssl-mod-1(config-ssl-proxy)# inservice
ssl-mod-1(config-ssl-proxy)#
ssl-mod-1(config-ssl-proxy)# exit
ssl-mod-1(config)#
ssl-mod-1(config)# ssl-proxy service ssl-offload-checkout
ssl-mod-1(config-ssl-proxy)# virtual ipaddr 8.100.100.111 protocol tcp port 443 secondary
ssl-mod-1(config-ssl-proxy)# server ipaddr 24.0.0.2 protocol tcp port 80
ssl-mod-1(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-mod-1(config-ssl-proxy)# nat client client-nat
ssl-mod-1(config-ssl-proxy)# inservice
ssl-mod-1(config-ssl-proxy)# exit
ssl-mod-1(config)#
ssl-mod-1(config)# ssl-proxy vlan 8
ssl-mod-1(config-vlan)# ipaddr 8.100.100.200 255.0.0.0
ssl-mod-1(config-vlan)# gateway 8.100.100.100
ssl-mod-1(config-vlan)# route 24.0.0.0 255.0.0.0 gateway 8.100.100.100
ssl-mod-1(config-vlan)# standby 1 ip 8.100.100.126
ssl-mod-1(config-vlan)# standby 1 timers 1 3
ssl-mod-1(config-vlan)# standby 1 priority 90
ssl-mod-1(config-vlan)# standby 1 preempt delay minimum 60
ssl-mod-1(config-vlan)# standby 2 ip 8.100.100.127
ssl-mod-1(config-vlan)# standby 2 timers 1 3
ssl-mod-1(config-vlan)# standby 2 priority 110
ssl-mod-1(config-vlan)# standby 2 preempt delay minimum 60
ssl-mod-1(config-vlan)# exit
ssl-mod-1(config)#
```

SSL 2 Configuration

This example shows how to configure the proxy services and the VLAN on SSL 2:

```
ssl-mod-2# config t
Enter configuration commands, one per line.  End with CNTL/Z.
ssl-mod-2(config)# ssl-proxy natpool client-nat 8.100.2.1 8.100.2.8 netmask 255.0.0.0
ssl-mod-2(config)# ssl-proxy service ssl-offload
ssl-mod-2(config-ssl-proxy)# virtual ipaddr 8.100.100.110 protocol tcp port 443 secondary
ssl-mod-2(config-ssl-proxy)# server ipaddr 24.0.0.2 protocol tcp port 80
ssl-mod-2(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-mod-2(config-ssl-proxy)# nat client client-nat
ssl-mod-2(config-ssl-proxy)# inservice
ssl-mod-2(config-ssl-proxy)# exit
ssl-mod-2(config)#
ssl-mod-2(config)# ssl-proxy service ssl-offload-checkout
ssl-mod-2(config-ssl-proxy)# virtual ipaddr 8.100.100.111 protocol tcp port 443 secondary
ssl-mod-2(config-ssl-proxy)# server ipaddr 24.0.0.2 protocol tcp port 80
ssl-mod-2(config-ssl-proxy)# certificate rsa general-purpose trustpoint cert
ssl-mod-2(config-ssl-proxy)# nat client client-nat
ssl-mod-2(config-ssl-proxy)# inservice
ssl-mod-2(config-ssl-proxy)# exit
ssl-mod-2(config)#
ssl-mod-2(config)# ssl-proxy vlan 8
ssl-mod-2(config-vlan)# ipaddr 8.100.100.250 255.0.0.0
ssl-mod-2(config-vlan)# gateway 8.100.100.100
ssl-mod-2(config-vlan)# route 24.0.0.0 255.0.0.0 gateway 8.100.100.100
ssl-mod-2(config-vlan)# standby priority 110
ssl-mod-2(config-vlan)# standby 1 ip 8.100.100.126
ssl-mod-2(config-vlan)# standby 1 timers 1 3
ssl-mod-2(config-vlan)# standby 1 priority 110
ssl-mod-2(config-vlan)# standby 1 preempt delay minimum 60
ssl-mod-2(config-vlan)# standby 2 ip 8.100.100.127
ssl-mod-2(config-vlan)# standby 2 timers 1 3
ssl-mod-2(config-vlan)# standby 2 priority 90
ssl-mod-2(config-vlan)# standby 2 preempt delay minimum 60
ssl-mod-2(config-vlan)# exit
ssl-mod-2(config)#
```

Displaying Statistics

These examples show how to display statistics to show that load balancing is occurring in two SSL Services Module:

SSL 1

```
ssl-mod-1# show ssl-proxy stats service
Service ssl-offload SSL Statistics:
    conns attempted     :0            conns completed     :0
    full handshakes     :0            resumed handshakes  :0
    conns in handshake  :0            conns in data       :0
    renegs attempted    :0            conns in reneg      :0
    blocks encrypted    :0            bytes encrypted     :0
    blocks decrypted    :0            bytes decrypted     :0
    valid cache entry   :0            session limit exceed:0
    handshake failures  :0            data failures       :0
    fatal alerts rcvd   :0            fatal alerts sent   :0
    bad macs received   :0            pad errors          :0
    no-cipher alerts    :0            no-compress alerts  :0
    ver mismatch alerts :0
```

```
Service ssl-offload-checkout SSL Statistics:
    conns attempted     :3288          conns completed     :3286
    full handshakes     :3287          resumed handshakes  :0
    conns in handshake  :1             conns in data       :1
    renegs attempted    :0             conns in reneg      :0
    blocks encrypted    :41468         bytes encrypted     :57831402
    blocks decrypted    :3287          bytes decrypted     :289256
    valid cache entry   :253152        session limit exceed:0
    handshake failures  :0             data failures       :0
    fatal alerts rcvd   :0             fatal alerts sent   :0
    bad macs received   :0             pad errors          :0
    no-cipher alerts    :0             no-compress alerts  :0
    ver mismatch alerts :0

ssl-mod-1# show standby
Ethernet0/0.8 - Group 1
  State is Standby
    7 state changes, last state change 00:03:37
  Virtual IP address is 8.100.100.126
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.004 secs
  Preemption enabled, delay min 60 secs
  Active router is 8.100.100.250, priority 110 (expires in 2.000 sec)
  Standby router is local
  Priority 90 (configured 90)
  IP redundancy name is "hsrp-Et0/0.8-1" (default)
Ethernet0/0.8 - Group 2
  State is Active
    2 state changes, last state change 01:53:29
  Virtual IP address is 8.100.100.127
  Active virtual MAC address is 0000.0c07.ac02
    Local virtual MAC address is 0000.0c07.ac02 (default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.648 secs
  Preemption enabled, delay min 60 secs
  Active router is local
  Standby router is 8.100.100.250, priority 90 (expires in 2.644 sec)
  Priority 110 (configured 110)
  IP redundancy name is "hsrp-Et0/0.8-2" (default)
ssl-mod-1#
```

SSL 2

```
ssl-mod-2# show ssl-proxy stats service
Service ssl-offload SSL Statistics:
    conns attempted     :4128          conns completed     :4126
    full handshakes     :4127          resumed handshakes  :0
    conns in handshake  :1             conns in data       :1
    renegs attempted    :0             conns in reneg      :0
    blocks encrypted    :51757         bytes encrypted     :72085513
    blocks decrypted    :4127          bytes decrypted     :363176
    valid cache entry   :136076        session limit exceed:0
    handshake failures  :0             data failures       :0
    fatal alerts rcvd   :0             fatal alerts sent   :0
    bad macs received   :0             pad errors          :0
    no-cipher alerts    :0             no-compress alerts  :0
    ver mismatch alerts :0
```

```
Service ssl-offload-checkout SSL Statistics:
    conns attempted     :0             conns completed     :0
    full handshakes     :0             resumed handshakes  :0
    conns in handshake  :0             conns in data       :3
    renegs attempted    :0             conns in reneg      :0
    blocks encrypted    :0             bytes encrypted     :0
    blocks decrypted    :0             bytes decrypted     :0
    valid cache entry   :126001        session limit exceed:0
    handshake failures  :0             data failures       :0
    fatal alerts rcvd   :0             fatal alerts sent   :0
    bad macs received   :0             pad errors          :0
    no-cipher alerts    :0             no-compress alerts  :0
    ver mismatch alerts :0

ssl-mod-2# show standby
Ethernet0/0.8 - Group 1
  State is Active
    2 state changes, last state change 02:23:54
  Virtual IP address is 8.100.100.126
  Active virtual MAC address is 0000.0c07.ac01
    Local virtual MAC address is 0000.0c07.ac01 (default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.232 secs
  Preemption enabled, delay min 60 secs
  Active router is local
  Standby router is 8.100.100.200, priority 90 (expires in 2.232 sec)
  Priority 110 (configured 110)
  IP redundancy name is "hsrp-Et0/0.8-1" (default)
Ethernet0/0.8 - Group 2
  State is Standby
    10 state changes, last state change 00:03:34
  Virtual IP address is 8.100.100.127
  Active virtual MAC address is 0000.0c07.ac02
    Local virtual MAC address is 0000.0c07.ac02 (default)
  Hello time 1 sec, hold time 3 sec
    Next hello sent in 0.876 secs
  Preemption enabled, delay min 60 secs
  Active router is 8.100.100.200, priority 110 (expires in 2.876 sec)
  Standby router is local
  Priority 90 (configured 90)
  IP redundancy name is "hsrp-Et0/0.8-2" (default)
ssl-mod-2#
```