



CHAPTER 26

Configuring Advanced Firewall Protection

This chapter describes how to prevent network attacks by configuring protection features, and includes the following sections:

- [Configuring Connection Settings and TCP State Bypass, page 26-1](#)
- [Using PISA to Permit or Deny Application Types, page 26-5](#)
- [Configuring the Fragment Size, page 26-11](#)
- [Configuring Anti-Spoofing, page 26-12](#)
- [Configuring TCP Options, page 26-12](#)
- [Configuring Global Timeouts, page 26-14](#)



Note

For Sun RPC server and encrypted traffic inspection settings, which you configure in the Configuration > Firewall > Advanced area (along with many of the topics in this chapter), see [Chapter 23, “Configuring Application Layer Protocol Inspection.”](#)

Configuring Connection Settings and TCP State Bypass

This section describes how to set maximum TCP and UDP connections, connection timeouts, and how to disable TCP sequence randomization. This section also describes how to enable TCP state bypass.

This section includes the following topics:

- [TCP Sequence Randomization Overview, page 26-1](#)
- [TCP State Bypass Overview, page 26-2](#)
- [Configuring Connection Settings and TCP State Bypass, page 26-4](#)

TCP Sequence Randomization Overview

Each TCP connection has two ISNs: one generated by the client and one generated by the server. The FWSM randomizes the ISN of the TCP SYN passing in both the inbound and outbound directions.

Randomizing the ISN of the protected host prevents an attacker from predicting the next ISN for a new connection and potentially hijacking the new session.

TCP initial sequence number randomization can be disabled if required. For example:

- If another in-line firewall is also randomizing the initial sequence numbers, there is no need for both firewalls to be performing this action, even though this action does not affect the traffic.
- If you use eBGP multi-hop through the FWSM, and the eBGP peers are using MD5. Randomization breaks the MD5 checksum.
- You use a WAAS device that requires the FWSM not to randomize the sequence numbers of connections.

**Note**

Because of the way TCP sequence randomization is implemented, if you enable Xlate Bypass (see the [“Enabling Xlate Bypass” section on page 21-17](#)), then disabling TCP sequence randomization only works for control connections, and not data connections; for data connections, the TCP sequence continues to be randomized.

TCP State Bypass Overview

This section describes how to use TCP state bypass, and includes the following sections:

- [Allowing Outbound and Inbound Flows through Separate FWSMs, page 26-2](#)
- [Unsupported Features, page 26-3](#)
- [Compatibility with NAT, page 26-3](#)
- [Connection Timeout, page 26-3](#)

Allowing Outbound and Inbound Flows through Separate FWSMs

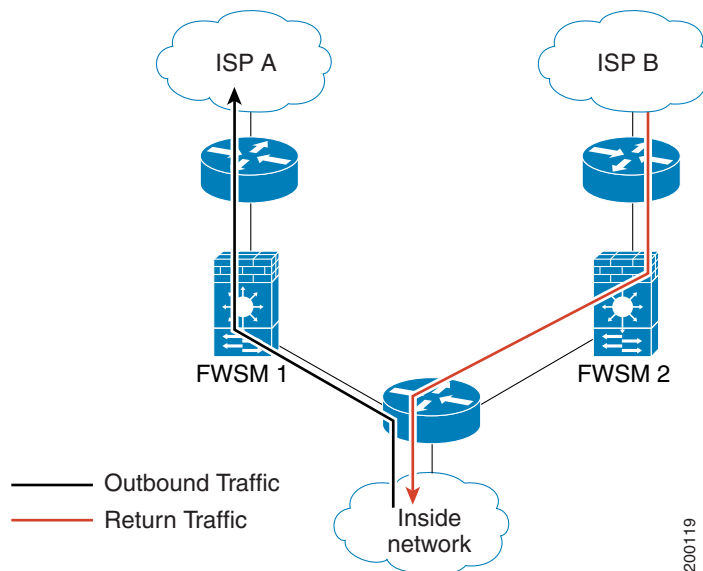
By default, all traffic that goes through the FWSM is inspected using the Adaptive Security Algorithm and is either allowed through or dropped based on the security policy. The FWSM maximizes the firewall performance by checking the state of each packet (is this a new connection or an established connection?) and assigning it to either the session management path (a new connection SYN packet), the fast path (an established connection), or the control plane path (advanced inspection).

TCP packets that match existing connections in the fast path can pass through the FWSM without rechecking every aspect of the security policy. This feature maximizes performance. However, the method of establishing the session in the fast path using the SYN packet, and the checks that occur in the fast path (such as TCP sequence number), can stand in the way of asymmetrical routing solutions: both the outbound and inbound flow of a connection must pass through the same FWSM.

For example, a new connection goes to FWSM 1. The SYN packet goes through the session management path, and an entry for the connection is added to the fast path table. If subsequent packets of this connection go through FWSM 1, then the packets will match the entry in the fast path, and are passed through. But if subsequent packets go to FWSM 2, where there was not a SYN packet that went through

the session management path, then there is no entry in the fast path for the connection, and the packets are dropped. [Figure 26-1](#) shows an asymmetric routing example where the outbound traffic goes through a different FWSM than the inbound traffic:

Figure 26-1 Asymmetric Routing



If you have asymmetric routing configured on upstream routers, and traffic alternates between two FWSMs, then you can configure TCP state bypass for specific traffic. TCP state bypass alters the way sessions are established in the fast path and disables the fast path checks. This feature treats TCP traffic much as it treats a UDP connection: when a non-SYN packet matching the specified networks enters the FWSM, and there is not a fast path entry, then the packet goes through the session management path to establish the connection in the fast path. Once in the fast path, the traffic bypasses the fast path checks.

Unsupported Features

The following features are not supported when you use TCP state bypass:

- Application inspection—Application inspection requires both inbound and outbound traffic to go through the same FWSM, so application inspection is not supported with TCP state bypass.
- AAA authenticated sessions—When a user authenticates with one FWSM, traffic returning via the other FWSM will be denied because the user did not authenticate with that FWSM.

Compatibility with NAT

Because the translation session is established separately for each FWSM, be sure to configure static NAT on both FWSMs for TCP state bypass traffic; if you use dynamic NAT, the address chosen for the session on FWSM 1 will differ from the address chosen for the session on FWSM 2.

Connection Timeout

If there is no traffic on a given connection for 2 minutes, the connection times out. You can override this default using the Properties > Timeouts > Connection check box. Normal TCP connections timeout by default after 60 minutes.

Configuring Connection Settings and TCP State Bypass



Note

You can also configure maximum connections and TCP sequence randomization in the NAT configuration. If you configure these settings for the same traffic using both methods, then the FWSM uses the lower limit. For TCP sequence randomization, if it is disabled using either method, then the FWSM disables TCP sequence randomization.

NAT also lets you configure embryonic connection limits, which triggers TCP Intercept to prevent a DoS attack. To configure connection limits, TCP randomization, and embryonic limits, see [Chapter 21, “Configuring NAT.”](#)

To configure connection settings, perform the following steps:

- Step 1** Configure a service policy on the Configuration > Firewall > Service Policy Rules pane according to [Chapter 22, “Configuring Service Policy Rules.”](#)

You can configure connection limits as part of a new service policy rule, or you can edit an existing service policy.



Note

In 3.x, when you configured connection settings for a service policy that matched an access list, then connection settings were applied to each individual ACE; in 4.0, connection settings are applied to the access list as a whole.

- Step 2** On the Rule Actions dialog box, click the **Connection Settings** tab.
- Step 3** To set the maximum number of simultaneous TCP and UDP connections for all clients in the traffic class, in the Maximum Connections area, enter a value in the **TCP & UDP Connections** field up to 65,536. The default is 0 for both protocols, which means the maximum possible connections are allowed.
- Step 4** To set the connection rate limit per second, in the Connection Rate Limit area, enter a value in the **Connection Rate Limit** field, between 0 and 65535. The default is 0, which means no limit on the connection rate.
- Step 5** To configure TCP timeouts, configure the following values in the TCP Timeout area:
- **Embryonic Connection Timeout**—Specifies the idle time until an embryonic connection slot is freed, between 0:0:1 and 0:4:15. The default is 0:0:20. You can also set this value to 0, which means the connection never times out.
 - **Half Closed Connection Timeout**—Specifies the idle time until a half closed connection slot is freed, between 0:0:1 and 0:4:15. The default is 0:0:20. You can also set this value to 0, which means the connection never times out. The FWSM does not send a reset when taking down half-closed connections.
 - **Connection Timeout**—Specifies the idle time until a connection slot is freed, between 0:5:0 and 1092:15:0. The default is 1:00:0. You can also set the value to 0, which means the connection never times out.
 - **Send reset to TCP endpoints before timeout**—Specifies that the FWSM should send a TCP reset message to the endpoints of the connection before freeing the connection slot.

**Note**

The Idle Timeout value (which applies to all protocols) has replaced the Connection Timeout value (which applies to TCP connections only). However, if your configuration includes the Connection Timeout value, it is still accepted. If you configure both values for the same class, then the Idle Timeout value is used instead of the Connection Timeout value when the traffic class does not *specifically* match TCP traffic. If the traffic class matches an access list that specifies TCP traffic explicitly, then the Connection Timeout value is used instead of the Idle Timeout value for the TCP traffic; other traffic that matches the access list uses the Idle Timeout value. (To match TCP traffic explicitly, when configuring the traffic class, choose Source and Destination IP Address (uses ACL), and specify TCP in the Protocol and Service area. To add additional access control entries (ACEs) that are not TCP-specific, add a new rule to the same interface and choose **Add rule to existing traffic class**.)

Step 6 To disable randomized sequence numbers, uncheck **Randomize Sequence Number**.

See the “[TCP Sequence Randomization Overview](#)” section on page 26-1 for more information.

Step 7 To configure the timeout for idle connections for all protocols, enter a value in the **Idle Timeout** field between 0:5:0 and 1092:15:0. The default is 1:00:00. You can also set the value to Unlimited (0:00:00), which means the connection never times out.

**Note**

The Idle Timeout value (which applies to all protocols) has replaced the Connection Timeout value (which applies to TCP connections only). However, if your configuration includes the Connection Timeout value, it is still accepted. If you configure both values for the same class, then the Idle Timeout value is used instead of the Connection Timeout value when the traffic class does not *specifically* match TCP traffic. If the traffic class matches an access list that specifies TCP traffic explicitly, then the Connection Timeout value is used instead of the Idle Timeout value for the TCP traffic; other traffic that matches the access list uses the Idle Timeout value. (To match TCP traffic explicitly, when configuring the traffic class, choose Source and Destination IP Address (uses ACL), and specify TCP in the Protocol and Service area. To add additional access control entries (ACEs) that are not TCP-specific, add a new rule to the same interface and choose **Add rule to existing traffic class**.)

Step 8 To enable TCP state bypass, in the Advanced Options area, check **TCP State Bypass**.

Step 9 Click **OK** or **Finish**.

Step 10 Click **Apply** to apply the configuration change.

Using PISA to Permit or Deny Application Types

**Note**

This feature depends on Cisco IOS Release 12.2(18)ZYA, and will not be supported until the Cisco IOS software is released.

The Programmable Intelligent Services Accelerator (PISA) on the switch supervisor can quickly determine the application type of a given flow by performing deep packet inspection. This determination can be made even if the traffic is not using standard ports. The FWSM can leverage the high-performance deep packet inspection of the PISA card so that it can permit or deny traffic based on the application type. Unlike the FWSM inspection feature, which passes through the control plane path, traffic that the

PISA tags can pass through the FWSM accelerated path. Another benefit of FWSM and PISA integration is to consolidate your security configuration on a single FWSM instead of having to configure multiple upstream switches with PISAs installed.

You might want to deny certain types of application traffic when you want to preserve bandwidth for critical application types. For example, you might deny the use of peer-to-peer (P2P) applications if they are affecting your other critical applications.

This section includes the following topics:

- [PISA Integration Overview, page 26-6](#)
- [Configuring the FWSM to Deny PISA Traffic, page 26-7](#)
- [Configuring the Switch for PISA/FWSM Integration, page 26-7](#)
- [Monitoring PISA Connections, page 26-10](#)

PISA Integration Overview

This section describes how the PISA works with the FWSM, and includes the following topics:

- [PISA Integration Guidelines and Limitations, page 26-6](#)
- [Using GRE for Tagging, page 26-6](#)
- [Failover Support, page 26-7](#)

PISA Integration Guidelines and Limitations

The following guidelines and limitations apply to PISA integration:

- The PISA and the FWSM cannot be in the same switch chassis. You can, however, use multiple PISAs upstream and downstream of the FWSM if desired.
- There is a slight performance impact on the PISA for traffic sent to the FWSM, due to the need to tag the packets for the FWSM (see the [“Using GRE for Tagging”](#) section.)
- When a UDP packet is denied due to the FWSM service policy, the corresponding session is not immediately deleted. Instead, it is allowed to time out, and the packets that hit this session in the meantime are dropped.
- It is possible for an end-user application to use the special GRE key that is used between the FWSM and the PISA. In such instances, the PISA generates a syslog message and drops these packets.
- The PISA takes several packets to determine the application type; therefore a session starts to be established on the FWSM before the PISA tagging commences. When the PISA tagging commences, the FWSM security policy is then applied, and if the policy is to deny the flow, the session is prevented from completing.
- For fragmented packets, the PISA tags the first fragment, and the FWSM reassembles the packet and acts upon it based on the encapsulation included in the first fragment.

See also the [“PISA Limitations and Restrictions”](#) section on page 26-8.

Using GRE for Tagging

After the PISA identifies the application used by a given traffic flow, it encapsulates all packets using GRE and includes a tag informing the FWSM of the application type. In addition, an outer IP header almost identical (except for the Layer 4 protocol, which now indicates GRE) to the inner/original IP

header is added. The original Layer 2 header is maintained. This preserves the original routing/switching paths for the modified packet. The GRE encapsulation adds 32 bytes (20 bytes for the outer IP header and 12 bytes for the GRE header).

After the FWSM receives the packet and acts on the information, it strips the GRE encapsulation from the packet.

When you configure the FWSM to deny traffic based on the PISA encapsulation, for the VLAN on which that traffic resides, the PISA encapsulates all traffic (including traffic that you did not specify for denial).

The GRE encapsulation increases the packet size slightly, so you should increase the MTU between the PISA and the FWSM according to the [“Changing the MTU on the Switch to Support Longer Packet Length” section on page 26-8](#).

The GRE encapsulation causes a slight performance impact for PISA traffic sent to the FWSM.

Failover Support

Failover of the PISA is independent of failover of the FWSM. If you have Stateful Failover on the FWSM, then the session information is maintained across the failover.

Configuring the FWSM to Deny PISA Traffic

To identify traffic that you want to deny using PISA tagging, perform the following steps:

-
- Step 1** Configure a service policy on the Configuration > Firewall > Service Policy Rules pane according to [Chapter 22, “Configuring Service Policy Rules.”](#)
You can configure this feature as part of a new service policy rule, or you can edit an existing service policy.
 - Step 2** On the Rule Actions dialog box, click the **Pisa Action** radio button.
You cannot configure PISA and application inspection for the same traffic.
 - Step 3** Click the **Pisa Action** tab.
 - Step 4** Set the actions for each protocol type. By default, all protocols are permitted unless you click the **Deny** radio button next to a protocol (the Default Action radio button is equivalent to the Permit setting).
If you want to set the default action to deny all protocols, then click **Deny all the protocols except the below permitted protocols** (the Default Action radio button is equivalent to the Deny setting) and then click **Permit** for the protocols you want to allow.
 - Step 5** Click **OK** or **Finish**.
 - Step 6** Click **Apply** to apply the configuration change.
-

Configuring the Switch for PISA/FWSM Integration

This section describes how to configure the switch for PISA/FWSM integration. This section describes how to use the Cisco IOS CLI; you cannot use ASDM to configure the switch for this feature. This section includes the following topics:

- [PISA Limitations and Restrictions, page 26-8](#)

- [Changing the MTU on the Switch to Support Longer Packet Length, page 26-8](#)
- [Configuring Classification on the PISA, page 26-8](#)
- [Configuring Tagging on the PISA, page 26-9](#)
- [Sample Switch Configurations for PISA Integration, page 26-9](#)

PISA Limitations and Restrictions

The following limitations and restrictions apply to the PISA:

- Network Based Application Recognition (NBAR) does not work on Layer 3 EtherChannels. Layer 2 EtherChannels are supported.
- The RP on the PISA does not support protocol tagging. So any packets going to the FWSM from the RP will not be tagged.
- NBAR implementation does not support IPv6. So protocol discovery and tagging are only applicable to IPv4. In addition to this restriction imposed by NBAR, the underlying PISA infrastructure also does not support acceleration of IPv6 packets.
- Currently there is a caveat in the L2 PISA implementation for VLANs that have been PISA-accelerated on an Layer 2 port (for example, a trunk); the SVI interfaces for VLANs passing through the accelerated Layer 2 port cannot be in an up state (they will become admin down).
- Multi-VLAN access ports are not supported.

See also the [“PISA Integration Guidelines and Limitations” section on page 26-6](#).

Changing the MTU on the Switch to Support Longer Packet Length

Because of the GRE encapsulation, you should increase the MTU size on VLANs used between the PISA and the FWSM. The GRE encapsulation adds 32 bytes (20 bytes for the outer IP header and 12 bytes for the GRE header).

- To change the MTU on a routed switch port or a Layer 3 interface (SVI), enter the following command:

```
Router(config-if)# mtu mtu_size
```

For an SVI, the *mtu_size* is between 64 and 9216 bytes. For a routed switch port, the *mtu_size* is between 1500 and 9216 bytes. The default MTU size is 1500 bytes.

- To configure the global LAN port MTU size for Layer 2 ports, enter the following command:

```
Router(config)# system jumbomtu mtu_size
```

The *mtu_size* can be between 1500 and 921 bytes. The default size is 9216 bytes.

Configuring Classification on the PISA

- To enable classification on a Layer 2 switch port (access, trunk or EtherChannel configured on a physical port) or a Layer 3 interface (SVI, routed port, or subinterface), enter the following command in interface configuration mode.

```
Router(config-if)# ip nbar protocol-discovery
```

- To show protocol discovery statistics on a Layer 2 or Layer 3 interface, enter the following command:

```
Router# show ip nbar protocol-discovery interface ifname
```

Configuring Tagging on the PISA

After protocol discovery is enabled, enable egress packet tagging by entering the following commands.



Note

Classification and tagging need to be enabled on the same port; for example, you cannot enable classification on access ports and tagging on a trunk port.

- To enable tagging on a switch port (access port) or a Layer 3 interface (SVI, routed port, or subinterface), enter the following command in interface configuration mode:

```
Router(config-if)# ip nbar protocol-tagging
```

- To enable tagging on a trunk port, enter the following command in interface configuration mode:

```
Router(config-if)# ip nbar protocol-tagging [vlan-list vlan-list]
```

Where the **vlan-list** *vlan-list* argument specifies a list of VLANs to be tagged. If not specified, all active VLANs are tagged.

The following commands help you monitor the tagging by the PISA:

- The following command displays tagging configuration information:

```
Router# show ip nbar protocol-tagging {key | interface ifname | summary}
```

Where the **key** keyword shows the GRE key used for the tagging.

The **interface** *ifname* argument shows if tagging is enabled on an interface.

The **summary** keyword shows all interfaces with tagging enabled.

- The following command shows the mapping of protocol name to ID:

```
Router# show ip nbar protocol-id [protocol_name]
```

If you enter the *protocol_name*, the mapped ID is shown. When omitted, the complete list of protocol names and IDs is shown.

- To show the number of packets tagged on the PISA, enter the following command:

```
Router# show platform pisa np tx counters
```

For example:

```
Router# show platform pisa np tx counters
```

```
TX Statistics(ME1)
-----
Errors: 0
.....
TX NBAR Protocol tagged pkt: 9869
```

Sample Switch Configurations for PISA Integration

Example 26-1 Layer 3 Mode (Interface-based, Routed port/SVI)

```
Router(config)# interface vlan 100
```

```

Router(config-if)# ip nbar protocol-discovery
! enables discovery
Router(config-if)# ip nbar protocol-tagging
! enables tagging
Router(config-if)# mtu 9216
! Allows packet sizes up to 9216 bytes without fragmenting

```

Example 26-2 Layer 2 Mode (Interface-based, Protocol Discovery on Uplink Ports)

```

Router(config)# interface gigabitethernet 6/1
Router(config-if)# ip nbar protocol-discovery
! Classification
Router(config-if)# ip nbar protocol-tagging vlan-list 100
! Tagging
Router(config-if)# mtu 9216
! Allow packet size up to 9216 bytes without fragmenting
Router(config)# system jumbo mtu 9216
! Set global LAN port MTU to 9216 bytes

```

Monitoring PISA Connections

This section includes the following topics:

- [Syslog Message for Dropped Connections, page 26-10](#)
- [Viewing PISA Connections on the FWSM, page 26-10](#)

Syslog Message for Dropped Connections

Syslog message 302014 (for TCP) and 302016 (for UDP) display when a PISA connection is denied. For example:

```
%FWSM-6-302014: Teardown TCP connection 144547133155839947 for inside:10.1.1.12/33407 to
outside:209.165.201.10/21 duration 0:00:00 bytes 160 PISA denied protocol
```

Viewing PISA Connections on the FWSM

To monitor connections from the PISA, use the **show conn** command. Connections that are tagged by the PISA are listed in the output with the “p” flag. The following is sample output from the **show conn** command:

```

hostname# show conn
2 in use, 3 most used
  Network Processor 1 connections
TCP out 10.1.1.10:21 in 209.165.201.12:33406 idle 0:00:04 Bytes 1668 FLAGS - UOIp
  Network Processor 2 connections
UDP out 10.1.1.255:137 in 10.1.1.11:137 idle 0:00:48 Bytes 288 FLAGS -
Multicast sessions:
  Network Processor 1 connections
  Network Processor 2 connections
IPv6 connections:
...

```

Configuring the Fragment Size

By default, the FWSM allows up to 24 fragments per IP packet, and up to 200 fragments awaiting reassembly. You might need to let fragments on your network if you have an application that routinely fragments packets, such as NFS over UDP. However, if you do not have an application that fragments traffic, we recommend that you do not allow fragments through the FWSM. Fragmented packets are often used as DoS attacks.

To modify the IP fragment database parameters of an interface, perform the following steps:

-
- Step 1** From the Configuration > Firewall > Advanced > Fragment pane, choose the interface to change in the Fragment table and click **Edit**. The Edit Fragment dialog box appears.
- Step 2** In the Edit Fragment dialog box, set the following parameters:
- **Size**—Sets the maximum number of packets that can be in the IP reassembly database waiting for reassembly. The default is 200.
 - **Chain Length**—Specifies the maximum number of packets into which a full IP packet can be fragmented. The default is 24 packets.
 - **Timeout**—Specifies the maximum number of seconds to wait for an entire fragmented packet to arrive. The timer starts after the first fragment of a packet arrives. If all fragments of the packet do not arrive by the number of seconds specified, all fragments of the packet that were already received will be discarded. The default is 5 seconds.
- Step 3** Click **OK**.
- Step 4** Click **Apply** in the Fragment pane.
- Step 5** To view the current IP fragment database statistics for each interface of the FWSM, click **Show Fragment** (see the [“View the Fragment Statistics”](#) section on page 26-11).
-

View the Fragment Statistics

The Show Fragment dialog box displays the operational data of the IP fragment reassembly module.

Fields

- **Size**—*Display only*. Displays the number of packets in the IP reassembly database waiting for reassembly. The default is 200.
- **Chain**—*Display only*. Displays the number of packets into which a full IP packet can be fragmented. The default is 24 packets.
- **Timeout**—*Display only*. Displays the number of seconds to wait for an entire fragmented packet to arrive. The timer starts after the first fragment of a packet arrives. If all fragments of the packet do not arrive by the number of seconds displayed, all fragments of the packet that were already received will be discarded. The default is 5 seconds.
- **Threshold**—*Display only*. Displays the IP packet threshold, or the limit after which no new chains can be created in the reassembly module.
- **Queue**—*Display only*. Displays the number of IP packets waiting in the queue for reassembly.
- **Assembled**—*Display only*. Displays the number of IP packets successfully reassembled.
- **Fail**—*Display only*. Displays the number of failed reassembly attempts.

- Overflow—*Display only*. Displays the number of IP packets in the overflow queue.

Configuring Anti-Spoofing

The Anti-Spoofing window lets you enable Unicast Reverse Path Forwarding on an interface. Unicast RPF guards against IP spoofing (a packet uses an incorrect source IP address to obscure its true source) by ensuring that all packets have a source IP address that matches the correct source interface according to the routing table.

Normally, the FWSM only looks at the destination address when determining where to forward the packet. Unicast RPF instructs the FWSM to also look at the source address; this is why it is called Reverse Path Forwarding. For any traffic that you want to allow through the FWSM, the FWSM routing table must include a route back to the source address. See RFC 2267 for more information.

For outside traffic, for example, the FWSM can use the default route to satisfy the Unicast RPF protection. If traffic enters from an outside interface, and the source address is not known to the routing table, the FWSM uses the default route to correctly identify the outside interface as the source interface.

If traffic enters the outside interface from an address that is known to the routing table, but is associated with the inside interface, then the FWSM drops the packet. Similarly, if traffic enters the inside interface from an unknown source address, the FWSM drops the packet because the matching route (the default route) indicates the outside interface.

Unicast RPF is implemented as follows:

- ICMP packets have no session, so each packet is checked.
- UDP and TCP have sessions, so the initial packet requires a reverse route lookup. Subsequent packets arriving during the session are checked using an existing state maintained as part of the session. Non-initial packets are checked to ensure they arrived on the same interface used by the initial packet.

Fields

- Interface—Lists the interface names.
- Anti-Spoofing Enabled—Shows whether an interface has Unicast RPF enabled, Yes or No.
- Enable—Enables Unicast RPF for the selected interface.
- Disable—Disables Unicast RPF for the selected interface.

Configuring TCP Options

The TCP Options pane lets you set parameters for TCP connections.

Fields

- Inbound and Outbound Reset—Sets whether to reset denied TCP connections for inbound and outbound traffic.
 - Interface—Shows the interface name.

- Inbound Reset—Shows the interface reset setting for inbound TCP traffic, Yes or No. Enabling this setting causes the FWSM to send TCP resets for all inbound TCP sessions that attempt to transit the FWSM and are denied by the FWSM based on access lists or AAA settings. Traffic between same security level interfaces is also affected. When this option is not enabled, the FWSM silently discards denied packets.
- Outbound Reset—Shows the interface reset setting for outbound TCP traffic, Yes or No. Enabling this setting causes the FWSM to send TCP resets for all outbound TCP sessions that attempt to transit the FWSM and are denied by the FWSM based on access lists or AAA settings. Traffic between same security level interfaces is also affected. When this option is not enabled, the FWSM silently discards denied packets.
- Edit—Sets the inbound and outbound reset settings for the interface.
- Other Options—Sets additional TCP options.
 - Send Reset Reply for Denied Outside TCP Packets—Enables resets for TCP packets that terminate at the least secure interface and are denied by the FWSM based on access lists or AAA settings. When this option is not enabled, the FWSM silently discards denied packets. If you enable Inbound Resets for the least secure interface (see [TCP Reset Settings](#)), then you do not also have to enable this setting; Inbound Resets handle to-the-FWSM traffic as well as through the FWSM traffic.
 - Force Maximum Segment Size for TCP—Sets the maximum TCP segment size in bytes, between 48 and any maximum number. The default value is 1380 bytes. You can disable this feature by setting the bytes to 0. Both the host and the server can set the maximum segment size when they first establish a connection. If either maximum exceeds the value you set here, then the FWSM overrides the maximum and inserts the value you set. For example, if you set a maximum size of 1200 bytes, when a host requests a maximum size of 1300 bytes, then the FWSM alters the packet to request 1200 bytes.
 - Force Minimum Segment Size for TCP—Overrides the maximum segment size to be no less than the number of bytes you set, between 48 and any maximum number. This feature is disabled by default (set to 0). Both the host and the server can set the maximum segment size when they first establish a connection. If either maximum is less than the value you set for the Force Minimum Segment Size for TCP Proxy field, then the FWSM overrides the maximum and inserts the “minimum” value you set (the minimum value is actually the smallest maximum allowed). For example, if you set a minimum size of 400 bytes, if a host requests a maximum value of 300 bytes, then the FWSM alters the packet to request 400 bytes.
 - Force TCP Connection to Linger in TIME_WAIT State for at Least 15 Seconds—Forces each TCP connection to linger in a shortened TIME_WAIT state of at least 15 seconds after the final normal TCP close-down sequence. You might want to use this feature if an end host application default TCP terminating sequence is a simultaneous close. The default behavior of the FWSM is to track the shutdown sequence and release the connection after two FINs and the ACK of the last FIN segment. This quick release heuristic enables the FWSM to sustain a high connection rate, based on the most common closing sequence, known as the normal close sequence. However, in a simultaneous close, both ends of the transaction initiate the closing sequence, as opposed to the normal close sequence where one end closes and the other end acknowledges prior to initiating its own closing sequence (see RFC 793). Thus, in a simultaneous close, the quick release forces one side of the connection to linger in the CLOSING state. Having many sockets in the CLOSING state can degrade the performance of an end host. For example, some WinSock mainframe clients are known to exhibit this behavior and degrade the performance of the mainframe server. Using this feature creates a window for the simultaneous close down sequence to complete.

TCP Reset Settings

This dialog box sets the inbound and outbound reset settings for an interface.

Fields

- **Send Reset Reply for Denied Inbound TCP Packets**—Sends TCP resets for all inbound TCP sessions that attempt to transit the FWSM and are denied by the FWSM based on access lists or AAA settings. Traffic between same security level interfaces is also affected. When this option is not enabled, the FWSM silently discards denied packets.

You might want to explicitly send resets for inbound traffic if you need to reset identity request (IDENT) connections. When you send a TCP RST (reset flag in the TCP header) to the denied host, the RST stops the incoming IDENT process so that you do not have to wait for IDENT to time out. Waiting for IDENT to time out can cause traffic to slow because outside hosts keep retransmitting the SYN until the IDENT times out, so the **service resetinbound** command might improve performance.

- **Send Reset Reply for Denied Outbound TCP Packets**—Sends TCP resets for all outbound TCP sessions that attempt to transit the FWSM and are denied by the FWSM based on access lists or AAA settings. Traffic between same security level interfaces is also affected. When this option is not enabled, the FWSM silently discards denied packets. This option is enabled by default. You might want to disable outbound resets to reduce the CPU load during traffic storms, for example.

Configuring Global Timeouts

The Timeouts pane lets you set the timeout durations for use with the FWSM. All durations are displayed in the format hh:mm:ss. It sets the idle time for the connection and translation slots of various protocols. If the slot has not been used for the idle time specified, the resource is returned to the free pool. TCP connection slots are freed approximately 60 seconds after a normal connection close sequence.



Note

It is recommended that you do not change these values unless advised to do so by Customer Support.

Fields

In all cases, except for Authentication absolute and Authentication inactivity, unchecking the check boxes means there is no timeout value. For those two cases, clearing the check box means to reauthenticate on every new connection.

- **Connection**—Modifies the idle time until a connection slot is freed. Enter 0:0:0 to disable timeout for the connection. This duration must be at least 5 minutes. The default is 1 hour.
- **Half-closed**—Modifies the idle time until a TCP half-closed connection closes. The minimum is 5 minutes. The default is 10 minutes. Enter 0:0:0 to disable timeout for a half-closed connection.
- **UDP**—Modifies the idle time until a UDP protocol connection closes. This duration must be at least 1 minute. The default is 2 minutes. Enter 0:0:0 to disable timeout.
- **ICMP**—Modifies the idle time after which general ICMP states are closed.
- **H.323**—Modifies the idle time until an H.323 media connection closes. The default is 5 minutes. Enter 0:0:0 to disable timeout.

- H.225—Modifies the idle time until an H.225 signaling connection closes. The H.225 default timeout is 1 hour (01:00:00). Setting the value of 00:00:00 means never close this connection. To close this connection immediately after all calls are cleared, a value of 1 second (00:00:01) is recommended.
- MGCP—Modifies the timeout value for MGCP which represents the idle time after which MGCP media ports are closed. The MGCP default timeout is 5 minutes (00:05:00). Enter 0:0:0 to disable timeout.
- MGCP PAT—Modifies the idle time after which an MGCP PAT translation is removed. The default is 5 minutes (00:05:00). The minimum time is 30 seconds. Uncheck the check box to return to the default value.
- SUNRPC—Modifies the idle time until a SunRPC slot is freed. This duration must be at least 1 minute. The default is 10 minutes. Enter 0:0:0 to disable timeout.
- SIP—Modifies the idle time until an SIP signalling port connection closes. This duration must be at least 5 minutes. The default is 30 minutes.
- SIP Media—Modifies the idle time until an SIP media port connection closes. This duration must be at least 1 minute. The default is 2 minutes.
- SIP Invite—Modifies the idle time after which pinholes for PROVISIONAL responses and media xlates will be closed. The minimum value is 0:1:0, the maximum value is 0:30:0. The default value is 0:03:00.
- SIP Disconnect—Modifies the idle time after which SIP session is deleted if the 200 OK is not received for a CANCEL or a BYE message. The minimum value is 0:0:1, the maximum value is 0:10:0. The default value is 0:02:00.
- Authentication absolute—Modifies the duration until the authentication cache times out and you have to reauthenticate a new connection. This duration must be shorter than the Translation Slot value. The system waits until you start a new connection to prompt you again. Enter 0:0:0 to disable caching and reauthenticate on every new connection.



Note Do not set this value to 0:0:0 if passive FTP is used on the connections.

- Authentication inactivity—Modifies the idle time until the authentication cache times out and users have to reauthenticate a new connection. This duration must be shorter than the Translation Slot value.
- Translation Slot—Modifies the idle time until a translation slot is freed. This duration must be at least 1 minute. The default is 3 hours. Enter 0:0:0 to disable timeout.

