CHAPTER 15

inspect ctiqbe through inspect xdmcp Commands

inspect ctiqbe

To enable CTIQBE protocol inspection, use the **inspect ctiqbe** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To disable inspection, use the **no** form of this command.

inspect ctiqbe

no inspect ctiqbe

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

The **inspect ctiqbe** command enables CTIQBE protocol inspection, which supports NAT, PAT, and bidirectional NAT. This enables Cisco IP SoftPhone and other Cisco TAPI/JTAPI applications to work successfully with Cisco CallManager for call setup across the FWSM.

The Telephony Application Programming Interface (TAPI) and Java Telephony Application Programming Interface (JTAPI) are used by many Cisco VoIP applications. Computer Telephony Interface Quick Buffer Encoding (CTIQBE) is used by Cisco TAPI Service Provider (TSP) to communicate with Cisco CallManager.

The following summarizes limitations that apply when using CTIQBE application inspection:

- CTIQBE application inspection does not support configurations using the **alias** command.
- Stateful Failover of CTIQBE calls is *not* supported.
- Using the **debug ctiqbe** command may delay message transmission, which may have a performance impact in a real-time environment. When you enable this debugging or logging and Cisco IP SoftPhone seems unable to complete call setup through the FWSM, increase the timeout values in the Cisco TSP settings on the system running Cisco IP SoftPhone.
- CTIQBE application inspection does *not* support CTIQBE messages fragmented in multiple TCP packets.

The following summarizes special considerations when using CTIQBE application inspection in specific scenarios:

- If two Cisco IP SoftPhones are registered with different Cisco CallManagers, which are connected to different interfaces of the FWSM, calls between these two phones will fail.

- When Cisco CallManager is located on the higher security interface compared to Cisco IP SoftPhones, if NAT or outside NAT is required for the Cisco CallManager IP address, the mapping must be static as Cisco IP SoftPhone requires the Cisco CallManager IP address to be specified explicitly in its Cisco TSP configuration on the PC.
- When using PAT or Outside PAT, if the Cisco CallManager IP address is to be translated, its TCP port 2748 must be statically mapped to the **same port** of the PAT (interface) address for Cisco IP SoftPhone registrations to succeed. The CTIQBE listening port (TCP 2748) is fixed and is not user-configurable on Cisco CallManager, Cisco IP SoftPhone, or Cisco TSP.

Inspecting Signaling Messages

For inspecting signaling messages, the **inspect ctiqbe** command often needs to determine locations of the media endpoints (for example, IP phones).

This information is used to prepare access-control and NAT state for media traffic to traverse the firewall transparently without manual configuration.

In determining these locations, the **inspect ctiqbe** command does **not** use the tunnel default gateway route. A tunnel default gateway route is a route of the form **route** *interface* **0 0** *metric* **tunneled**. This route overrides the default route for packets that egress from IPSec tunnels. Therefore, if the **inspect ctiqbe** command is desired for VPN traffic, do not configure the tunnel default gateway route. Instead, us other static routing or dynamic routing.

Examples

You enable the CTIQBE inspection engine as shown in the following example, which creates a class map to match CTIQBE traffic on the default port (2748). The service policy is then applied to the outside interface.

```
hostname(config)# class-map ctiqbe-port
hostname(config-cmap)# match port tcp eq 2748
hostname(config-cmap)# exit
hostname(config)# policy-map ctiqbe_policy
hostname(config-pmap)# class ctiqbe-port
hostname(config-pmap-c)# inspect ctiqbe
hostname(config-pmap-c)# exit
hostname(config)# service-policy ctiqbe_policy interface outside
```

To enable CTIQBE inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
show conn	Displays the connection state for different connection types.
show ctiqbe	Displays information regarding the CTIQBE sessions established across the FWSM. Displays information about the media connections allocated by the CTIQBE inspection engine.
timeout	Sets the maximum idle time duration for different protocols and session types.

inspect dcerpc

To enable inspection of DCERPC traffic destined for the endpoint-mapper, use the **inspect dcerpc** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect dcerpc [*map_name*]

no inspect dceprc [*map_name*]

Syntax Description

map_name	(Optional) The name of the DCERPC map.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.2(1)	This command was introduced.

Usage Guidelines

The **inspect dcerpc** command enables or disables application inspection for the DCERPC protocol.

Examples

The following example shows how to define a DCERPC inspection policy map with the timeout configured for DCERPC pinholes and enforce the endpoint mapper service during binding so that only its service traffic is processed. The **lookup-operation** keyword enables the lookup operation of the endpoint mapper service.

```
hostname(config)# policy-map type inspect dcerpc dcerpc_map
hostname(config-pmap)# timeout pinhole 0:10:00
hostname(config-pmap)# endpoint-mapper epm-service-only lookup-operation

hostname(config)# class-map dcerpc
hostname(config-cmap)# match port tcp eq 135

hostname(config)# policy-map global-policy
hostname(config-pmap)# class dcerpc
hostname(config-pmap-c)# inspect dcerpc dcerpc_map

hostname(config)# service-policy global-policy global
```

Related Commands

Commands	Description
class	Identifies a class map name in the policy map.
class-map type inspect	Creates an inspection class map to match traffic specific to an application.
policy-map	Creates a Layer 3/4 policy map.
show running-config policy-map	Display all current policy map configurations.
timeout pinhole	Configures the timeout for DCERPC pinholes and overrides the global system pinhole timeout.

inspect dns

To enable DNS inspection (if it has been previously disabled), use the **inspect dns** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. Use the **inspect dns** command to specify the maximum DNS packet length. To disable DNS inspection, use the **no** form of this command.

inspect dns [**maximum-length** *max_pkt_length*]

no inspect dns [**maximum-length** *max_pkt_length*]

Syntax Description

maximum-length	(Optional) Specifies the maximum DNS packet length. The default is 512. If you enter the **inspect dns** command without the **maximum-length** option, DNS packet size is not checked.
max_pkt_length	The maximum DNS packet length. Longer packets will be dropped.

Defaults

This command is enabled by default.

The default **maximum-length** for the DNS packet size is 512.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	**Routed**	**Transparent**	**Single**	**Context**	**System**
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol dns** command, which is now deprecated.

Usage Guidelines

DNS guard tears down the DNS session associated with a DNS query as soon as the DNS reply is forwarded by the FWSM. DNS guard also monitors the message exchange to ensure that the ID of the DNS reply matches the ID of the DNS query.

When DNS inspection is enabled, which it is the default, the FWSM performs the following additional tasks:

- Translates the DNS record based on the configuration completed using the **alias**, **static** and **nat** commands (DNS rewrite). Translation only applies to the A-record in the DNS reply. Therefore, reverse lookups, which request the PTR record, are not affected by DNS rewrite.

Note DNS rewrite is not applicable for PAT because multiple PAT rules are applicable for each A-record and the PAT rule to use is ambiguous.

- Enforces the maximum DNS message length (the default is 512 bytes and the maximum length is 65535 bytes). Reassembly is performed as necessary to verify that the packet length is less than the maximum length configured. The packet is dropped if it exceeds the maximum length.

Note If you enter the **inspect dns** command without the **maximum-length** option, DNS packet size is not checked

- Enforces a domain-name length of 255 bytes and a label length of 63 bytes.
- Verifies the integrity of the domain-name referred to by the pointer if compression pointers are encountered in the DNS message.
- Checks to see if a compression pointer loop exists.

A single connection is created for multiple DNS sessions, as long as they are between the same two hosts, and the sessions have the same 5-tuple (source/destination IP address, source/destination port, and protocol). DNS identification is tracked by *app_id*, and the idle timer for each app_id runs independently.

Because the app_id expires independently, a legitimate DNS response can only pass through the FWSM within a limited period of time and there is no resource build-up. However, if you enter the **show conn** command, you will see the idle timer of a DNS connection being reset by a new DNS session. This is due to the nature of the shared DNS connection and is by design.

How DNS Rewrite Works

When DNS inspection is enabled, DNS rewrite provides full support for NAT of DNS messages originating from any interface.

If a client on an inside network requests DNS resolution of an inside address from a DNS server on an outside interface, the DNS A-record is translated correctly. If the DNS inspection engine is disabled, the A-record is not translated.

DNS rewrite performs two functions:

- Translating a public address (the routable or “mapped” address) in a DNS reply to a private address (the “real” address) when the DNS client is on a private interface.
- Translating a private address to a public address when the DNS client is on the public interface.

As long as DNS inspection remains enabled, you can configure DNS rewrite using the **alias**, **static**, or **nat** commands. For details about the syntax and function of these commands, refer to the appropriate command page.

Examples

The following example changes the maximum DNS packet length to 1500 bytes. Although DNS inspection is enabled by default, you still need to create a traffic map to identify DNS traffic and then apply the policy map to the appropriate interface.

```
hostname(config)# class-map dns-port
hostname(config-cmap)# match port udp eq 53
hostname(config-cmap)# exit
hostname(config)# policy-map sample_policy
hostname(config-pmap)# class dns-port
```

```
hostname(config-pmap-c)# inspect dns maximum-length 1500
hostname(config-pmap-c)# exit
hostname(config)# service-policy sample_policy interface outside
```

To change the maximum DNS packet length for all interfaces, use the **global** parameter in place of **interface outside**.

The following example shows how to disable DNS:

```
hostname(config)# policy-map sample_policy
hostname(config-pmap)# class dns-port
hostname(config-pmap-c)# no inspect dns
hostname(config-pmap-c)# exit
hostname(config)# service-policy sample_policy interface outside
```

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug dns	Enables debug information for DNS.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect esmtp

To enable extended SMTP application inspection, use the **inspect esmtp** command in class configuration mode. The class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect esmtp

no inspect esmtp

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

ESMTP application inspection provides improved protection against SMTP-based attacks by restricting the types of SMTP commands that can pass through the FWSM and by adding monitoring capabilities.

Extended SMTP application inspection, as enabled by the **inspect esmtp** command, occurs in control plane path processing; therefore, it occurs on the single, general purpose processor on the FWSM.

ESMTP is an enhancement to the SMTP protocol and is similar is most respects to SMTP. For convenience, the term SMTP is used in this document to refer to both SMTP and ESMTP. The application inspection process for extended SMTP is similar to SMTP application inspection and includes support for SMTP sessions. Most commands used in an extended SMTP session are the same as those used in an SMTP session but an ESMTP session is considerably faster and offers more options related to reliability and security, such as delivery status notification.

The **inspect esmtp** command includes the functionality provided by the **inspect smtp** command, and provides additional support for some extended SMTP commands. Extended SMTP application inspection adds support for extended SMTP commands including AUTH, EHLO, ETRN, HELP, SAML, SEND, SOML, STARTLS, and VRFY. Along with the support for seven RFC 821 commands (DATA, HELO, MAIL, NOOP, QUIT, RCPT, RSET), the FWSM supports a total of fifteen SMTP commands.

Other extended SMTP commands, such as ATRN, ONEX, VERB, CHUNKING, and private extensions and are not supported. Unsupported commands are translated into Xs, which are rejected by the internal server. This results in a message such as "500 Command unknown: 'XXX'." Incomplete commands are discarded.

Note If a policy map contains both the **inspect smtp** command and the **inspect esmtp** command, only the first command listed in the policy map is applied to matching traffic.

The **inspect esmtp** command changes the characters in the server SMTP banner to asterisks except for the "2", "0", "0" characters. Carriage return (CR) and linefeed (LF) characters are ignored.

With SMTP inspection enabled, a Telnet session used for interactive SMTP may hang if the following rules are not observed: SMTP commands must be at least four characters in length; must be terminated with carriage return and line feed; and must wait for a response before issuing the next reply.

An SMTP server responds to client requests with numeric reply codes and optional human readable strings. SMTP application inspection controls and reduces the commands that the user can use as well as the messages that the server returns. SMTP inspection performs three primary tasks:

- Restricts SMTP requests to seven basic SMTP commands and eight extended commands.
- Monitors the SMTP command-response sequence.
- Generates an audit trail—Audit record 108002 is generated when invalid character embedded in the mail address is replaced. For more information, see RFC 821.

SMTP inspection monitors the command and response sequence for the following anomalous signatures:

- Truncated commands.
- Incorrect command termination (not terminated with <CR><LR>).
- The MAIL and RCPT commands specify who are the sender and the receiver of the mail. Mail addresses are scanned for strange characters. The pipeline character | is deleted (changed to a blank space) and | are only allowed if they are used to define a mail address | must be preceded by "<").
- Unexpected transition by the SMTP server.
- For unknown commands, the FWSM changes all the characters in the packet to X. In this case, the server will generate an error code to the client. Because of the change in the packet, the TCP checksum has to be recalculated or adjusted.
- TCP stream editing.
- Command pipelining.

Note FWSM supports SMTP and Extended SMTP Inspection for inbound traffic only; namely, FWSM supports inspection of traffic from a lower security level to a higher security level.

Examples

You enable the SMTP inspection engine as shown in the following example, which creates a class map to match SMTP traffic on the default port (25). The service policy is then applied to the outside interface.

```
hostname(config)# class-map smtp-port
hostname(config-cmap)# match port tcp eq 25
hostname(config-cmap)# exit
hostname(config)# policy-map smtp_policy
hostname(config-pmap)# class smtp-port
hostname(config-pmap-c)# inspect esmtp
```

```
hostname(config-pmap-c)# exit
hostname(config)# service-policy smtp_policy interface outside
```

To enable SMTP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug smtp	Enables debug information for SMTP.
inspect smtp	Enables standard (non-extedned) SMTP application inspection.
policy-map	Associates a class map with specific security actions.
show conn	Displays the connection state for different connection types, including SMTP.

inspect ftp

To configure the port for FTP inspection or to enable enhanced inspection, use the **inspect ftp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect ftp [**strict** [*map_name*]]

no inspect ftp [**strict** [*map_name*]]

Syntax Description

map_name	The name of the FTP map.
strict	(Optional) Enables enhanced inspection of FTP traffic and forces compliance with RFC standards.

Caution Use caution when moving FTP to a higher port. For example, if you set the FTP port to 2021, all connections that initiate to port 2021 will have their data payload interpreted as FTP commands.

Defaults

The FWSM listens to port 21 for FTP by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol ftp** command, which is now deprecated.

Usage Guidelines

The FTP application inspection inspects the FTP sessions and performs four tasks:

- Prepares dynamic secondary data connections
- Tracks **ftp** command-response sequence
- Generates an audit trail
- NATs embedded IP addresses

FTP application inspection prepares secondary channels for FTP data transfer. The channels are allocated in response to a file upload, a file download, or a directory listing event and must be pre-negotiated. The port is negotiated through the PORT or PASV commands.

Note If you disable FTP inspection engines with the **no inspect ftp** command, outbound users can start connections only in passive mode, and all inbound FTP is disabled.

Using the strict Option

The **strict** option prevents web browsers from sending embedded commands in FTP requests. Each **ftp** command must be acknowledged before a new command is allowed. Connections sending embedded commands are dropped. The **strict** option only lets an FTP server generate the 227 command and only lets an FTP client generate the PORT command. The 227 and PORT commands are checked to ensure they do not appear in an error string.

Caution The use of the **strict** option may break FTP clients that do not comply with the RFC standards.

If the **strict** option is enabled, each **ftp** command and response sequence is tracked for the following anomalous activity:

- Truncated command—Number of commas in the PORT and PASV reply command is checked to see if it is five. If it is not five, then the PORT command is assumed to be truncated and the TCP connection is closed.
- Incorrect command—Checks the **ftp** command to see if it ends with <CR><LF> characters, as required by the RFC. If it does not, the connection is closed.
- Size of RETR and STOR commands—These are checked against a fixed constant. If the size is greater, then an error message is logged and the connection is closed.
- Command spoofing—The PORT command should always be sent from the client. The TCP connection is denied if a PORT command is sent from the server.
- Reply spoofing—PASV reply command (227) should always be sent from the server. The TCP connection is denied if a PASV reply command is sent from the client. This prevents the security hole when the user executes “227 xxxxx a1, a2, a3, a4, p1, p2.”
- TCP stream editing.
- Invalid port negotiation—The negotiated dynamic port value is checked to see if it is less than 1024. As port numbers in the range from 1 to 1024 are reserved for well-known connections, if the negotiated port falls in this range, then the TCP connection is freed.
- Command pipelining—The number of characters present after the port numbers in the PORT and PASV reply command is cross checked with a constant value of 8. If it is more than 8, then the TCP connection is closed.
- The FWSM replaces the FTP server response to the SYST command with a series of Xs. to prevent the server from revealing its system type to FTP clients. To override this default behavior, use the **no mask-syst-reply** command in FTP map configuration mode.

Note To identify specific FTP commands that are not permitted to pass through the FWSM, identify an FTP map and use the **request-command deny** command. For details, see the **ftp-map** and the **request-command deny** command pages.

FTP Log Messages

FTP application inspection generates the following log messages:

- An Audit record 302002 is generated for each file that is retrieved or uploaded.

- The **ftp** command is checked to see if it is RETR or STOR and the retrieve and store commands are logged.
- The username is obtained by looking up a table providing the IP address.
- The username, source IP address, destination IP address, NAT address, and the file operation are logged.
- Audit record 201005 is generated if the secondary dynamic channel preparation failed due to memory shortage.

In conjunction with NAT, the FTP application inspection translates the IP address within the application payload. This is described in detail in RFC 959.

Examples

The following example identifies FTP traffic, defines an FTP map, defines a policy, enables strict FTP inspection, and applies the policy to the outside interface:

```
hostname(config)# class-map ftp-port
hostname(config-cmap)# match port tcp eq 21
hostname(config-cmap)# exit
hostname(config)# ftp-map inbound_ftp
hostname(config-inbound_ftp)# request-command deny put stou appe
hostname(config-ftp-map)# exit
hostname(config)# policy-map inbound_policy
hostname(config-pmap)# class ftp-port
hostname(config-pmap-c)# inspect ftp strict inbound_ftp
hostname(config-pmap-c)# exit
hostname(config-pmap)# exit
hostname(config)# service-policy inbound_policy interface outside
```

To enable strict FTP application inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Note Only specify the port for the FTP control connection and not the data connection. The FWSM stateful inspection engine dynamically prepares the data connection as necessary.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
mask-syst-reply	Hides the FTP server response from clients.
policy-map	Associates a class map with specific security actions.
request-command deny	Specifies FTP commands to disallow.
service-policy	Applies a policy map to one or more interfaces.

inspect gtp

To enable or disable GTP inspection or to define a GTP map for controlling GTP traffic or tunnels, use the **inspect gtp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the command, use the **no** form of this command.

inspect gtp [*map_name*]

no inspect gtp [*map_name*]

Note GTP inspection requires a special license. If you enter the **inspect gtp** command on a FWSM without the required license, the FWSM displays an error message.

Syntax Description

map_name	(Optional) Name for the GTP map.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

GTP is the tunnelling protocol for GPRS, and helps provide secure access over wireless networks. GPRS is a data network architecture that is designed to integrate with existing GSM networks. It offers mobile subscribers uninterrupted, packet-switched data services to corporate networks and the Internet. For an overview of GTP, see the "Applying Application Layer Protocol Inspection" chapter in the *Catalyst 6500 Series Switch and Cisco 7600 Series Router Firewall Services Module Configuration Guide*.

Use the **gtp-map** command to identify a specific map to use for defining the parameters for GTP. When you enter this command, the system enters a configuration mode that lets you enter the different commands used for defining the specific map. The actions that you can specify for messages that fail the criteria set using the different configuration commands include **allow**, **reset**, or **drop**. In addition to these actions, you can specify to log the event or not.

After defining the GTP map, you use the **inspect gtp** command to enable the map. Then you use the **class-map**, **policy-map**, and **service-policy** commands to define a class of traffic, to apply the **inspect** command to the class, and to apply the policy to one or more interfaces.

The string **gtp**, used as a port value, is automatically converted to the port value 3386. The well-known ports for GTP are as follows:

- 3386
- 2123

The following features are not supported in 7.0:

- NAT, PAT, Outside NAT, alias, and Policy NAT
- Ports other than 3386, 2123, and 2152
- Validating the tunneled IP packet and its contents

Inspecting Signaling Messages

For inspecting signaling messages, the **inspect gtp** command often needs to determine locations of the media endpoints (for example, IP phones).

This information is used to prepare access-control and NAT state for media traffic to traverse the firewall transparently without manual configuration.

In determining these locations, the **inspect gtp** command does **not** use the tunnel default gateway route. A tunnel default gateway route is a route of the form **route** *interface* **0 0** *metric* **tunneled**. This route overrides the default route for packets that egress from IPSec tunnels. Therefore, if the **inspect gtp** command is desired for VPN traffic, do not configure the tunnel default gateway route. Instead, us other static routing or dynamic routing.

Examples

The following example shows how to use access lists to identify GTP traffic, define a GTP map, define a policy, and apply the policy to the outside interface:

```
hostname(config)# access-list gtp-acl permit udp any any eq 3386
hostname(config)# access-list gtp-acl permit udp any any eq 2123
hostname(config)# class-map gtp-traffic
hostname(config)# match access-list gtp-acl
hostname(config)# gtp-map gtp-policy
hostname(config)# policy-map inspection_policy
hostname(config-pmap)# class gtp-traffic
hostname(config-pmap-c)# inspect gtp gtp-policy
hostname(config)# service-policy inspection_policy interface outside
```

Note This example enables GTP inspection with the default values. To change the default values, refer to the **gtp-map** command page and to the command pages for each command that is entered from GTP map configuration mode.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
clear service-policy inspect gtp	Clears global GTP statistics.
debug gtp	Displays detailed information about GTP inspection.
service-policy	Applies a policy map to one or more interfaces.

inspect h323

To enable H.323 application inspection or to change the ports to which the FWSM listens, use the **inspect h323** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect h323 {**h225** [*h225_map*] | **ras**}

no inspect h323 {**h225** [*h225_map*] | **ras**}

Syntax Description

h225	Enables H.225 signalling inspection.
h225_map	(Optional) The name of an H.225 application inspection map, which defines the configuration required to use the FWSM in topologies involving Cisco HSI and H.323 endpoints'.
ras	Enables RAS inspection.

Defaults

The default port assignments are as follows:

- h323 h225 1720
- h323 ras 1718-1719

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	**Routed**	**Transparent**	**Single**	**Context**	**System**
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol h323** command, which is now deprecated.

Usage Guidelines

The **inspect h323** command provides support for H.323 compliant applications such as Cisco CallManager and VocalTec Gatekeeper. H.323 is a suite of protocols defined by the International Telecommunication Union (ITU) for multimedia conferences over LANs. The FWSM supports H.323 through Version 4, including the H.323 v3 feature Multiple Calls on One Call Signaling Channel.

With H.323 inspection enabled, the FWSM supports multiple calls on the same call signaling channel, a feature introduced with H.323 Version 3. This feature reduces call setup time and reduces the use of ports on the FWSM.

The two major functions of H.323 inspection are as follows:

- NAT the necessary embedded IPv4 addresses in the H.225 and H.245 messages. Because H.323 messages are encoded in PER encoding format, the FWSM uses an ASN.1 decoder to decode the H.323 messages.
- Dynamically allocate the negotiated H.245 and RTP/RTCP connections.

How H.323 Works

The H.323 collection of protocols collectively may use up to two TCP connection and four to six UDP connections. FastStart uses only one TCP connection, and RAS uses a single UDP connection for registration, admissions, and status.

An H.323 client may initially establish a TCP connection to an H.323 server using TCP port 1720 to request Q.931 call setup. As part of the call setup process, the H.323 terminal supplies a port number to the client to use for an H.245 TCP connection. The H.245 connection is for call negotiation and media channel setup. In environments where H.323 gatekeeper is in use, the initial packet is transmitted using UDP.

H.323 inspection monitors the Q.931 TCP connection to determine the H.245 port number. If the H.323 terminals are not using FastStart, the FWSM dynamically allocates the H.245 connection based on the inspection of the H.225 messages.

Note The H.225 connection can also be dynamically allocated when using RAS.

Within each H.245 message, the H.323 endpoints exchange port numbers that are used for subsequent UDP data streams. H.323 inspection inspects the H.245 messages to identify these ports and dynamically creates connections for the media exchange. Real-Time Transport Protocol (RTP) uses the negotiated port number, while RTP Control Protocol (RTCP) uses the next higher port number.

The H.323 control channel handles H.225 and H.245 and H.323 RAS. H.323 inspection uses the following ports.

- 1718—UDP port used for gatekeeper discovery
- 1719—UDP port used for RAS and for gatekeeper discovery
- 1720—TCP Control Port

If the ACF message from the gatekeeper goes through the FWSM, a pinhole will be opened for the H.225 connection. The H.245 signaling ports are negotiated between the endpoints in the H.225 signaling. When an H.323 gatekeeper is used, the FWSM opens an H.225 connection based on inspection of the ACF message. If the FWSM does not see the ACF message, you might need to open an access list for the well-known H.323 port 1720 for the H.225 call signaling.

The FWSM dynamically allocates the H.245 channel after inspecting the H.225 messages and then hooks up to the H.245 channel to be fixed up as well. That means whatever H.245 messages pass through the FWSM pass through the H.245 application inspection, NATing embedded IP addresses and opening the negotiated media channels.

The H.323 ITU standard requires that a TPKT header, defining the length of the message, precede the H.225 and H.245, before being passed on to the reliable connection. Because the TPKT header does not necessarily need to be sent in the same TCP packet as the H.225/H.245 message, the FWSM must remember the TPKT length to process/decode the messages properly. The FWSM keeps a data structure for each connection and that data structure contains the TPKT length for the next expected message.

If the FWSM needs to NAT any IP addresses, then it will have to change the checksum, the UUIE (user-user information element) length, and the TPKT, if included in the TCP packet with the H.225 message. If the TPKT is sent in a separate TCP packet, then the FWSM will proxy ACK that TPKT and append a new TPKT to the H.245 message with the new length.

Note The FWSM does not support TCP options in the Proxy ACK for the TPKT.

Each UDP connection with a packet going through H.323 inspection is marked as an H.323 connection and will time out with the H.323 timeout as configured using the **timeout** command.

Limitations and Restrictions

The following are some of the known issues and limitations when using H.323 application inspection:

- Static PAT may not properly translate IP addresses embedded in optional fields within H.323 messages. If you experience this kind of problem, do not use static PAT with H.323.
- It has been observed that when a NetMeeting client registers with an H.323 gatekeeper and tries to call an H.323 gateway that is also registered with the H.323 gatekeeper, the connection is established but no voice is heard in either direction. This problem is unrelated to the FWSM.
- If you configure a network static where the network static is the same as a third-party netmask and address, then any outbound H.323 connection fails.

Inspecting Signaling Messages

For inspecting signaling messages, the **inspect h323** command often needs to determine locations of the media endpoints (for example, IP phones).

This information is used to prepare access control and NAT state for media traffic to traverse the firewall transparently without manual configuration.

In determining these locations, the **inspect h323** command does **not** use the tunnel default gateway route. A tunnel default gateway route is a route of the form **route** *interface* **0 0** *metric* **tunneled**. This route overrides the default route for packets that egress from IPSec tunnels. Therefore, if the **inspect h323** command is desired for VPN traffic, do not configure the tunnel default gateway route. Instead, us other static routing or dynamic routing.

Using an H.225 Map

An H.225 map allows the FWSM to open dynamic, port-specific pinholes for an H.245 connection when an HSI is involved in H.225 call-signalling. The H.225 map provides information about the HSI and its associated endpoints, which is required to establish this connection without compromising the security of the network protected by the FWSM.

Table 15-1 summarizes the commands used to perform the required configuration:

Table 15-1 H..225 Configuration Commands

Command	Configuration mode	Description
h225-map	Global configuration mode	Defines an H.225 application inspection map and enables H.225 map configuration mode.One H225 map can contain a maximum of five HSI groups.
hsi-group	H.225 map configuration mode	Defines an HSI group and enables HSI group configuration mode. Each HSI group can contain a maximum of ten endpoints
hsi	HSI group configuration mode	Identifies the HSI.
endpoint	HSI group configuration mode	Identifies one or more endpoints within the HSI group.

Examples

The following example shows how to enable the H.323 inspection engine, which creates a class map to match H.323 traffic on the default port (1720). The service policy is then applied to the outside interface.

```
hostname(config)# class-map h323-port
hostname(config-cmap)# match port tcp eq 1720
hostname(config-cmap)# exit
hostname(config)# policy-map h323_policy
hostname(config-pmap)# class h323-port
hostname(config-pmap-c)# inspect h323
hostname(config-pmap-c)# exit
hostname(config)# service-policy h323_policy interface outside
```

To enable inspection for all interfaces, enter the **global** parameter in place of **interface outside**

The following example illustrates the H.225 configuration required when an FWSM interconnects H.323 endpoints and a Cisco CallManager must establish a connection between these endpoints:

```
hostname(config)# access-list h323_acl permit udp any any eq 1720
hostname(config)# access-list h323_acl permit udp any any eq 1721
hostname(config)# class-map h323-traffic
hostname(config-cmap)# match access-list h323_acl
hostname(config-cmap)# exit
hostname(config)# h225-map sample_map
hostname(config-h225-map-hsi-grp)# hsi 10.10.15.11
hostname(config-h225-map-hsi-grp)# endpoint 10.3.6.1 inside
hostname(config-h225-map-hsi-grp)# endpoint 10.10.25.5 outside
hostname(config-h225-map-hsi-grp)# exit
hostname(config)# policy-map sample_policy
hostname(config-pmap)# class h323_port
hostname(config-pmap-c)# inspect h323 ras
hostname(config-pmap-c)# inspect h323 h225 sample_map
hostname(config-pmap-c)# exit
hostname(config)# service-policy sample_policy interface outside
```

Related Commands

Commands	Description
debug h323	Enables the display of debug information for H.323.
show h225	Displays information for H.225 sessions established across the FWSM.
show h245	Displays information for H.245 sessions established across the FWSM by endpoints using slow start.
show h323-ras	Displays information for H.323 RAS sessions established across the FWSM.
timeout	Configures idle time after which an H.225 signalling connection or an H.323 control connection will be closed.

inspect http

To enable HTTP application inspection or to change the ports to which the FWSM listens, use the **inspect http command** in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect http [*map_name*]

no inspect http [*map_name*]

Syntax Description

map_name	(Optional) The name of the HTTP map.

Defaults

The default port for HTTP is 80.

Enhanced HTTP inspection is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol http** command, which is now deprecated.

Usage Guidelines

The **inspect http** command protects against specific attacks and other threats that may be associated with HTTP traffic. HTTP inspection performs several functions:

- Enhanced HTTP inspection
- Java and ActiveX filtering

The second feature is configured in conjunction with the **filter** command.

Enhanced HTTP inspection verifies that HTTP messages conform to RFC 2616, use RFC-defined methods or supported extension methods, and comply with various other criteria. In many cases, you can configure these criteria and the system response when the criteria are not met. The actions that you can specify for messages that fail the criteria set using the different configuration commands include **allow**, **reset**, or **drop**. In addition to these actions, you can specify to log the event or not.

The criteria that you can apply to HTTP messages include the following:

- Does not include any method on a configurable list.
- Specific transfer encoding method or application type.

- HTTP transaction adheres to RFC specification.
- Message body size is within configurable limits.
- Request and response message header size is within a configurable limit.
- URI length is within a configurable limit.
- The content-type in the message body matches the header.
- The content-type in the response message matches the *accept-type* field in the request message.
- The content-type in the message is included in a predefined internal list.
- Message meets HTTP RFC format criteria.
- Presence or absence of selected supported applications.
- Presence or absence of selected encoding types.

Note The actions that you can specify for messages that fail the criteria set using the different configuration commands include **allow**, **reset**, or **drop**. In addition to these actions, you can specify to log the event or not.

To enable enhanced HTTP inspection, enter the **inspect http** *http-map* command. The rules that this applies to HTTP traffic are defined by the specific HTTP map, which you configure by entering the **http-map** command and HTTP map configuration mode commands.

Note When you enable HTTP inspection with an HTTP map, strict HTTP inspection with the action reset and log is enabled by default. You can change the actions performed in response to inspection failure, but you cannot disable strict inspection as long as the HTTP map remains enabled.

Examples

The following example shows how to identify HTTP traffic, define an HTTP map, define a policy, and apply the policy to the outside interface:

```
hostname(config)# class-map http-port
hostname(config-cmap)# match port tcp eq 80
hostname(config-cmap)# exit
hostname(config)# http-map inbound_http
hostname(config-http-map)# content-length min 100 max 2000 action reset log
hostname(config-http-map)# content-type-verification match-req-rsp reset log
hostname(config-http-map)# max-header-length request bytes 100 action log reset
hostname(config-http-map)# max-uri-length 100 action reset log
hostname(config-http-map)# exit
hostname(config)# policy-map inbound_policy
hostname(config-pmap)# class http-port
hostname(config-pmap-c)# inspect http inbound_http
hostname(config-pmap-c)# exit
hostname(config-pmap)# exit
hostname(config)# service-policy inbound_policy interface outside
```

This example causes the FWSM to reset the connection and create a syslog entry when it detects any traffic that contain the following:

- Messages less than 100 bytes or exceeding 2000 bytes
- Unsupported content types

- HTTP headers exceeding 100 bytes
- URIs exceeding 100 bytes

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug appfw	Displays detailed information about HTTP application inspection.
debug http-map	Displays detailed information about traffic associated with an HTTP map.
http-map	Defines an HTTP map for configuring enhanced HTTP inspection.
policy-map	Associates a class map with specific security actions.

inspect icmp

To configure the ICMP inspection engine, use the **inspect icmp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect icmp

no inspect icmp

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol icmp** command, which is now deprecated.

Usage Guidelines

The ICMP inspection engine allows ICMP traffic to be inspected like TCP and UDP traffic. Without the ICMP inspection engine, we recommend that you do not allow ICMP through the FWSM in an ACL. Without stateful inspection, ICMP can be used to attack your network. The ICMP inspection engine ensures that there is only one response for each request, and that the sequence number is correct

When ICMP inspection is disabled, which is the default configuration, ICMP echo reply messages are denied from a lower security interface to a higher security interface, even if it is in response to an ICMP echo request.

Examples

The following example shows how to enable the ICMP application inspection engine, which creates a class map to match ICMP traffic using the ICMP protocol ID, which is 1 for IPv4 and 58 for IPv6. The service policy is then applied to the outside interface.

```
hostname(config)# class-map icmp-class
hostname(config-cmap)# match default-inspection-traffic
hostname(config-cmap)# exit
hostname(config)# policy-map icmp_policy
hostname(config-pmap)# class icmp-class
hostname(config-pmap-c)# inspect icmp
hostname(config-pmap-c)# exit
hostname(config)# service-policy icmp_policy interface outside
```

To enable ICMP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
icmp	Configures access rules for ICMP traffic that terminates at a FWSM interface.
policy-map	Defines a policy that associates security actions with one or more traffic classes.
service-policy	Applies a policy map to one or more interfaces.

inspect icmp error

To enable application inspection for ICMP error messages, use the **inspect icmp error** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect icmp error

no inspect icmp error

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol icmp error** command, which is now deprecated.

Usage Guidelines

Use the **inspect icmp error** command to create xlates for intermediate hops that send ICMP error messages, based on the static/NAT configuration. By default, the security appliance hides the IP addresses of intermediate hops. However, using the **inspect icmp error** command makes the intermediate hop IP addresses visible. The FWSM overwrites the packet with the translated IP addresses.

Note When you configure the **inspect icmp error** command for inspection of ICMP error messages, you must create an access list to permit the returning ICMP time-exceeded packets.

When enabled, the ICMP error inspection engine makes the following changes to the ICMP packet:

- In the IP Header, the NAT IP is changed to the Client IP (Destination Address and Intermediate Hop Address) and the IP checksum is modified.
- In the ICMP Header, the ICMP checksum is modified due to the changes in the ICMP packet.
- In the Payload, the following changes are made:
  - Original packet NAT IP is changed to the Client IP
  - Original packet NAT port is changed to the Client Port
  - Original packet IP checksum is recalculated

When an ICMP error message is retrieved, whether ICMP error inspection is enabled or not, the ICMP payload is scanned to retrieve the five-tuple (src ip , dest ip, src port, dest port, and ip protocol) from the original packet. A lookup is performed, using the retrieved five-tuple, to determine the original address of the client and to locate an existing session associated with the specific five-tuple. If the session is not found, the ICMP error message is dropped.

Examples

The following example shows how to enable the ICMP error application inspection engine, which creates a class map to match ICMP traffic using the ICMP protocol ID, which is 1 for IPv4 and 58 for IPv6. The service policy is then applied to the outside interface.

```
hostname(config)# class-map icmp-class
hostname(config-cmap)# match default-inspection-traffic
hostname(config-cmap)# exit
hostname(config)# policy-map icmp_policy
hostname(config-pmap)# class icmp-class
hostname(config-pmap-c)# inspect icmp error
hostname(config-pmap-c)# exit
hostname(config)# service-policy icmp_policy interface outside
```

To enable ICMP error inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
icmp	Configures access rules for ICMP traffic that terminates at a FWSM interface.
inspect icmp	Enables or disables the ICMP inspection engine.
policy-map	Defines a policy that associates security actions with one or more traffic classes.
service-policy	Applies a policy map to one or more interfaces.

inspect ils

To enable ILS application inspection or to change the ports to which the FWSM listens, use the **inspect ils** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect ils

no inspect ils

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol ils** command, which is now deprecated.

Usage Guidelines

The **inspect ils** command provides NAT support for Microsoft NetMeeting, SiteServer, and Active Directory products that use LDAP to exchange directory information with an ILS server.

Use the *port* option to change the default port assignment from 389. Use the *-port* option to apply ILS inspection to a range of port numbers.

The FWSM supports NAT for ILS, which is used to register and locate endpoints in the ILS or SiteServer Directory. PAT cannot be supported because only IP addresses are stored by an LDAP database.

For search responses, when the LDAP server is located outside, NAT should be considered to allow internal peers to communicate locally while registered to external LDAP servers. For such search responses, xlates are searched first, and then DNAT entries to obtain the correct address. If both of these searches fail, then the address is not changed. For sites using NAT 0 (no NAT) and not expecting DNAT interaction, we recommend that the inspection engine be turned off to provide better performance.

Additional configuration may be necessary when the ILS server is located inside the FWSM border. This would require a hole for outside clients to access the LDAP server on the specified port, typically TCP 389.

Because ILS traffic only occurs on the secondary UDP channel, the TCP connection is disconnected after the TCP inactivity interval. By default, this interval is 60 minutes and can be adjusted using the **timeout** command.

ILS/LDAP follows a client/server model with sessions handled over a single TCP connection. Depending on the client actions, several of these sessions may be created.

During connection negotiation time, a BIND PDU is sent from the client to the server. Once a successful BIND RESPONSE from the server is received, other operational messages may be exchanged (such as ADD, DEL, SEARCH, or MODIFY) to perform operations on the ILS Directory. The ADD REQUEST and SEARCH RESPONSE PDUs may contain IP addresses of NetMeeting peers, used by H.323 (SETUP and CONNECT messages) to establish the NetMeeting sessions. Microsoft NetMeeting v2.X and v3.X provides ILS support.

The ILS inspection performs the following operations:

- Decodes the LDAP REQUEST/RESPONSE PDUs using the BER decode functions
- Parses the LDAP packet
- Extracts IP addresses
- Translates IP addresses as necessary
- Encodes the PDU with translated addresses using BER encode functions
- Copies the newly encoded PDU back to the TCP packet
- Performs incremental TCP checksum and sequence number adjustment

ILS inspection has the following limitations:

- Referral requests and responses are not supported
- Users in multiple directories are not unified
- Single users having multiple identities in multiple directories cannot be recognized by NAT

Note Because H.225 call signalling traffic only occurs on the secondary UDP channel, the TCP connection is disconnected after the interval specified by the TCP **timeout** command. By default, this interval is set at 60 minutes.

Examples

The following example shows how to enable the ILS inspection engine, which creates a class map to match ILS traffic on the default port (389). The service policy is then applied to the outside interface.

```
hostname(config)# class-map ils-port
hostname(config-cmap)# match port tcp eq 389
hostname(config-cmap)# exit
hostname(config)# policy-map ils_policy
hostname(config-pmap)# class ils-port
hostname(config-pmap-c)# inspect ils
hostname(config-pmap-c)# exit
hostname(config)# service-policy ils_policy interface outside
```

To enable ILS inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug ils	Enables debug information for ILS.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect mgcp

To enable MGCP application inspection or to change the ports to which the FWSM listens, use the **inspect mgcp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect mgcp [*map_name*]

no inspect mgcp [*map_name*]

Syntax Description

map_name	(Optional) The name of the MGCP map.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol mgcp** command, which is now deprecated.

Usage Guidelines

To use MGCP, you usually need to configure at least two **inspect** commands: one for the port on which the gateway receives commands, and one for the port on which the Call Agent receives commands. Normally, a Call Agent sends commands to the default MGCP port for gateways, 2427, and a gateway sends commands to the default MGCP port for Call Agents, 2727.

MGCP is used for controlling media gateways from external call control elements called media gateway controllers or call agents. A media gateway is typically a network element that provides conversion between the audio signals carried on telephone circuits and data packets carried over the Internet or over other packet networks. Using NAT and PAT with MGCP lets you support a large number of devices on an internal network with a limited set of external (global) addresses.

Examples of media gateways are:

- Trunking gateways, that interface between the telephone network and a Voice over IP network. Such gateways typically manage a large number of digital circuits.
- Residential gateways, that provide a traditional analog (RJ11) interface to a Voice over IP network. Examples of residential gateways include cable modem/cable set-top boxes, xDSL devices, broad-band wireless devices.

- Business gateways, that provide a traditional digital PBX interface or an integrated soft PBX interface to a Voice over IP network.

MGCP messages are transmitted over UDP. A response is sent back to the source address (IP address and UDP port number) of the command, but the response may not arrive from the same address as the command was sent to. This can happen when multiple call agents are being used in a failover configuration and the call agent that received the command has passed control to a backup call agent, which then sends the response.

Note MGCP call agents send AUEP messages to determine if MGCP end points are present. This establishes a flow through the FWSM and allows MGCP end points to register with the call agent.

Use the **call-agent** and **gateway** commands in MGCP map configuration mode to configure the IP addresses of one or more call agents and gateways. Use the **command-queue** command in MGCP map configuration mode to specify the maximum number of MGCP commands that will be allowed in the command queue at one time.

Inspecting Signaling Messages

For inspecting signaling messages, the **inspect mgcp** command often needs to determine locations of the media endpoints (for example, IP phones).

This information is used to prepare access-control and NAT state for media traffic to traverse the firewall transparently without manual configuration.

Examples

The following example shows how to identify MGCP traffic, define a MGCP map, define a policy, and apply the policy to the outside interface. This creates a class map to match MGCP traffic on the default ports (2427 and 2727). The service policy is then applied to the outside interface.

```
hostname(config)# access-list mgcp_acl permit tcp any any eq 2427
hostname(config)# access-list mgcp_acl permit tcp any any eq 2727
hostname(config)# class-map mgcp_port
hostname(config-cmap)# match access-list mgcp_acl
hostname(config-cmap)# exit
hostname(config)# mgcp-map inbound_mgcp
hostname(config-mgcp-map)# call-agent 10.10.11.5 101
hostname(config-mgcp-map)# call-agent 10.10.11.6 101
hostname(config-mgcp-map)# call-agent 10.10.11.7 102
hostname(config-mgcp-map)# call-agent 10.10.11.8 102
hostname(config-mgcp-map)# gateway 10.10.10.115 101
hostname(config-mgcp-map)# gateway 10.10.10.116 102
hostname(config-mgcp-map)# gateway 10.10.10.117 102
hostname(config-mgcp-map)# command-queue 150
hostname(config-mgcp-map)# exit
hostname(config)# policy-map inbound_policy
hostname(config-pmap)# class mgcp_port
hostname(config-pmap-c)# inspect mgcp mgcp-map inbound_mgcp
hostname(config-pmap-c)# exit
hostname(config)# service-policy inbound_policy interface outside
```

This configuration allows call agents 10.10.11.5 and 10.10.11.6 to control gateway 10.10.10.115, and allows call agents 10.10.11.7 and 10.10.11.8 to control both gateways 10.10.10.116 and 10.10.10.117. The maximum number of MGCP commands that can be queued is 150.

To enable MGCP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug mgcp	Enables MGCP debug information.
mgcp-map	Defines an MGCP map and enables MGCP map configuration mode.
show mgcp	Displays information about MGCP sessions established through the FWSM.
timeout	Sets the maximum idle time duration for different protocols and session types.

inspect netbios

To enable NetBIOS application inspection or to change the ports to which the FWSM listens, use the **inspect netbios** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect netbios

no inspect netbios

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1	This command was introduced.

Usage Guidelines

The **inspect netbios** command enables or disables application inspection for the NetBIOS protocol.

Examples

The following example shows how to enable the NetBIOS inspection engine, which creates a class map to match NetBIOS traffic on the default UDP ports (137 and 138). The service policy is then applied to the outside interface.

```
hostname(config)# class-map netbios-port
hostname(config-cmap)# match port udp range 137 138
hostname(config-cmap)# exit
hostname(config)# policy-map netbios_policy
hostname(config-pmap)# class netbios-port
hostname(config-pmap-c)# inspect netbios
hostname(config-pmap-c)# exit
hostname(config)# service-policy netbios_policy interface outside
```

To enable NetBIOS inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect pptp

To enable PPTP application inspection or to change the ports to which the FWSM listens, use the **inspect pptp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

```
inspect pptp

no inspect pptp
```

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

The Point-to-Point Tunneling Protocol (PPTP) is a protocol for tunneling PPP traffic. A PPTP session is composed of one TCP channel and usually two PPTP GRE tunnels. The TCP channel is the control channel used for negotiating and managing the PPTP GRE tunnels. The GRE tunnels carries PPP sessions between the two hosts.

When enabled, PPTP application inspection inspects PPTP protocol packets and dynamically creates the GRE connections and xlates necessary to permit PPTP traffic. Only Version 1, as defined in RFC 2637, is supported.

PAT is only performed for the modified version of GRE (RFC 2637) when negotiated over the PPTP TCP control channel. Port Address Translation is *not* performed for the unmodified version of GRE (RFC 1701, RFC 1702).

Specifically, the FWSM inspects the PPTP version announcements and the outgoing call request/response sequence. Only PPTP Version 1, as defined in RFC 2637, is inspected. Further inspection on the TCP control channel is disabled if the version announced by either side is not Version 1. In addition, the outgoing-call request and reply sequence are tracked. Connections and xlates are dynamically allocated as necessary to permit subsequent secondary GRE data traffic.

The PPTP inspection engine must be enabled for PPTP traffic to be translated by PAT. Additionally, PAT is only performed for a modified version of GRE (RFC2637) and only if it is negotiated over the PPTP TCP control channel. PAT is not performed for the unmodified version of GRE (RFC 1701 and RFC 1702).

As described in RFC 2637, the PPTP protocol is mainly used for the tunneling of PPP sessions initiated from a modem bank PAC (PPTP Access Concentrator) to the headend PNS (PPTP Network Server). When used this way, the PAC is the remote client and the PNS is the server.

However, when used for VPN by Windows, the interaction is inverted. The PNS is a remote single-user PC that initiates connection to the head-end PAC to gain access to a central network.

Examples

The following example shows how to enable the PPTP inspection engine, which creates a class map to match PPTP traffic on the default port (1723). The service policy is then applied to the outside interface.

```
hostname(config)# class-map pptp-port
hostname(config-cmap)# match port tcp eq 1723
hostname(config-cmap)# exit
hostname(config)# policy-map pptp_policy
hostname(config-pmap)# class pptp-port
hostname(config-pmap-c)# inspect pptp
hostname(config-pmap-c)# exit
hostname(config)# service-policy pptp_policy interface outside
```

To enable PPTP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug pptp	Enables debug information for PPTP.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect rsh

To enable RSH application inspection or to change the ports to which the FWSM listens, use the **inspect rsh** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect rsh

no inspect rsh

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol rsh** command, which is now deprecated.

Usage Guidelines

The RSH protocol uses a TCP connection from the RSH client to the RSH server on TCP port 514. The client and server negotiate the TCP port number where the client listens for the STDERR output stream. RSH inspection supports NAT of the negotiated port number if necessary.

Examples

The following example shows how to enable the RSH inspection engine, which creates a class map to match RSH traffic on the default port (514). The service policy is then applied to the outside interface.

```
hostname(config)# class-map rsh-port
hostname(config-cmap)# match port tcp eq 514
hostname(config-cmap)# exit
hostname(config)# policy-map rsh_policy
hostname(config-pmap)# class rsh-port
hostname(config-pmap-c)# inspect rsh
hostname(config-pmap-c)# exit
hostname(config)# service-policy rsh_policy interface outside
```

To enable RSH inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect rtsp

To enable RTSP application inspection or to change the ports to which the FWSM listens, use the **inspect rtsp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect rtsp

no inspect rtsp

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol rtsp** command, which is now deprecated.

Usage Guidelines

The **inspect rtsp** command lets the FWSM pass RTSP packets. RTSP is used by RealAudio, RealNetworks, Apple QuickTime 4, RealPlayer, and Cisco IP/TV connections.

Note For Cisco IP/TV, use RTSP TCP port 554 and TCP 8554.

RTSP applications use the well-known port 554 with TCP (rarely UDP) as a control channel. The FWSM only supports TCP, in conformity with RFC 2326. This TCP control channel is used to negotiate the data channels that will be used to transmit audio/video traffic, depending on the transport mode that is configured on the client.

The supported RDT transports are: rtp/avp, rtp/avp/udp, x-real-rdt, x-real-rdt/udp, and x-pn-tng/udp.

The FWSM parses Setup response messages with a status code of 200. If the response message is travelling inbound, the server is outside relative to the FWSM and dynamic channels need to be opened for connections coming inbound from the server. If the response message is outbound, then the FWSM does not need to open dynamic channels.

Because RFC 2326 does not require that the client and server ports must be in the SETUP response message, the FWSM will need to keep state and remember the client ports in the SETUP message. QuickTime places the client ports in the SETUP message and then the server responds with only the server ports.

Using RealPlayer

To use RealPlayer, it is important to properly configure transport mode. For the FWSM, add an **access-list** command statement from the server to the client or vice versa. For RealPlayer, change transport mode by clicking **Options>Preferences>Transport>RTSP Settings**.

If using TCP mode on the RealPlayer, select the **Use TCP to Connect to Server** and **Attempt to use TCP for all content** check boxes. On the FWSM, there is no need to configure the inspection engine.

If using UDP mode on the RealPlayer, select the **Use TCP to Connect to Server** and **Attempt to use UDP for static content** check boxes, and for live content not available via Multicast. On the FWSM, add an **inspect rtsp** *port* command statement.

Restrictions and Limitations

The following restrictions apply to the **inspect rtsp** command:

- The FWSM does not support multicast RTSP or RTSP messages over UDP.
- The FWSM does not have the ability to recognize HTTP cloaking where RTSP messages are hidden in the HTTP messages.
- The FWSM cannot perform NAT on RTSP messages because the embedded IP addresses are contained in the SDP files as part of HTTP or RTSP messages. Packets could be fragmented and the FWSM cannot perform NAT on fragmented packets.
- With Cisco IP/TV, the number of NATs the FWSM performs on the SDP part of the message is proportional to the number of program listings in the Content Manager (each program listing can have at least six embedded IP addresses).
- You can configure NAT for Apple QuickTime 4 or RealPlayer. Cisco IP/TV only works with NAT if the Viewer and Content Manager are on the outside network and the server is on the inside network.
- Media streams delivered over HTTP are not supported by RTSP application inspection. This is because RTSP inspection does not support HTTP cloaking (RTSP wrapped in HTTP).

Examples

The following example shows how to enable the RTSP inspection engine, which creates a class map to match RTSP traffic on the default ports (554 and 8554). The service policy is then applied to the outside interface.

```
hostname(config)# access-list rtsp-acl permit tcp any any eq 554
hostname(config)# access-list rtsp-acl permit tcp any any eq 8554
hostname(config)# class-map rtsp-traffic
hostname(config-cmap)# match access-list rtsp-acl
hostname(config-cmap)# exit
hostname(config)# policy-map rtsp_policy
hostname(config-pmap)# class rtsp-traffic
hostname(config-pmap-c)# inspect rtsp
hostname(config-pmap-c)# exit
hostname(config)# service-policy rtsp_policy interface outside
```

To enable RTSP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug rtsp	Enables debug information for RTSP.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect sip

To enable SIP application inspection or to change the ports to which the FWSM listens, use the **inspect sip** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect sip [*map_name*]

no inspect sip [*map_name*]

Syntax Description

map_name	(Optional) The name of a SIP map created using the **sip-map** command. A SIP map lets you specify additional inspection parameters for SIP inspection, such as IP address privacy, which you can configure with the **ip-address-privacy** command in SIP map configuration mode.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol sip** command, which is now deprecated.

Usage Guidelines

SIP, as defined by the IETF, enables VoIP calls. SIP works with SDP for call signalling. SDP specifies the details of the media stream. Using SIP, the FWSM can support any SIP Voice over IP (VoIP) gateways and VoIP proxy servers. SIP and SDP are defined in the following RFCs:

- SIP: Session Initiation Protocol, RFC 2543
- SDP: Session Description Protocol, RFC 2327

To support SIP calls through the FWSM, signaling messages for the media connection addresses, media ports, and embryonic connections for the media must be inspected, because while the signaling is sent over a well-known destination port (UDP/TCP 5060), the media streams are dynamically allocated. Also, SIP embeds IP addresses in the user-data portion of the IP packet. SIP inspection applies NAT for these embedded IP addresses.

Note If a remote endpoint tries to register with a SIP proxy on a network protected by the FWSM, the registration will fail under very specific conditions. These conditions are when PAT is configured for the remote endpoint, the SIP registrar server is on the outside network, and the port is missing in the contact field in the REGISTER message sent by the endpoint to the proxy server.

Instant Messaging

Instant Messaging refers to the transfer of messages between users in near real-time. The MESSAGE/INFO methods and 202 Accept response are used to support IM as defined in the following RFCs:

- Session Initiation Protocol (SIP)-Specific Event Notification, RFC 3265
- Session Initiation Protocol (SIP) Extension for Instant Messaging, RFC 3428

MESSAGE/INFO requests can come in at any time after registration/subscription. For example, two users can be online at any time, but not chat for hours. Therefore, the SIP inspection engine opens pinholes, which will time out according to the configured SIP timeout value. This value must be configured at least five minutes longer than the subscription duration. The subscription duration is defined in the Contact Expires value and is typically 30 minutes.

Because MESSAGE/INFO requests are typically sent using a dynamically allocated port other than port 5060, they are required to go through the SIP inspection engine.

Note Only the Chat feature is currently supported. Whiteboard, File Transfer, and Application Sharing are not supported. RTC Client 5.0 is not supported.

Technical Details

SIP inspection NATs the SIP text-based messages, recalculates the content length for the SDP portion of the message, and recalculates the packet length and checksum. It dynamically opens media connections for ports specified in the SDP portion of the SIP message as address/ports on which the endpoint should listen.

SIP inspection has a database with indices CALL_ID/FROM/TO from the SIP payload that identifies the call, as well as the source and destination. Contained within this database are the media addresses and media ports that were contained in the SDP media information fields and the media type. There can be multiple media addresses and ports for a session. RTP/RTCP connections are opened between the two endpoints using these media addresses/ports.

The well-known port 5060 must be used on the initial call setup (INVITE) message. However, subsequent messages may not have this port number. The SIP inspection engine opens signaling connection pinholes, and marks these connections as SIP connections. This is done for the messages to reach the SIP application and be NATed.

As a call is set up, the SIP session is considered in the “transient” state. This state remains until a Response message is received indicating the RTP media address and port on which the destination endpoint is listening. If there is a failure to receive the response messages within one minute, the signaling connection will be torn down.

Once the final handshake is made, the call state is moved to active and the signaling connection will remain until a BYE message is received.

If an inside endpoint initiates a call to an outside endpoint, a media hole is opened to the outside interface to allow RTP/RTCP UDP packets to flow to the inside endpoint media address and media port specified in the INVITE message from the inside endpoint. Unsolicited RTP/RTCP UDP packets to an inside interface will not traverse the FWSM, unless the FWSM configuration specifically allows it.

The media connections are torn down within two minutes after the connection becomes idle. This is, however, a configurable timeout and can be set for a shorter or longer period of time.

Inspecting Signaling Messages

For inspecting signaling messages, the **inspect sip** command often needs to determine locations of the media endpoints (for example, IP phones).

This information is used to prepare access-control and NAT state for media traffic to traverse the firewall transparently without manual configuration.

In determining these locations, the **inspect sip** command does **not** use the tunnel default gateway route. A tunnel default gateway route is a route of the form **route** *interface* **0 0** *metric* **tunneled**. This route overrides the default route for packets that egress from IPSec tunnels. Therefore, if the **inspect sip** command is desired for VPN traffic, do not configure the tunnel default gateway route. Instead, use other static routing or dynamic routing.

Examples

The following example shows how to enable the SIP inspection engine, which creates a class map to match SIP traffic on the default port (5060). The service policy is then applied to the outside interface.

```
hostname(config)# class-map sip-port
hostname(config-cmap)# match port tcp eq 5060
hostname(config-cmap)# policy-map sip_policy
hostname(config-pmap)# class sip-port
hostname(config-pmap-c)# inspect sip
hostname(config-pmap-c)# service-policy sip_policy interface outside
hostname(config)#
```

To enable SIP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands=

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
policy-map	Associates a class map with specific security actions.
show sip	Displays information about SIP sessions established through the FWSM.
show conn	Displays the connection state for different connection types.
sip-map	Defines additional SIP inspection parameters.

inspect skinny

T o enable SCCP (Skinny) application inspection or to change the ports to which the FWSM listens, use the **inspect skinny** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect skinny

no inspect skinny

Syntax Description

This command has no arguments or keywords.

Defaults

This command is enabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol skinny** command, which is now deprecated.

Usage Guidelines

Skinny (or Simple) Client Control Protocol (SCCP) is a simplified protocol used in VoIP networks. Cisco IP Phones using SCCP can coexist in an H.323 environment. When used with Cisco CallManager, the SCCP client can interoperate with H.323-compliant terminals. Application layer functions in the FWSM recognize SCCP Version 3.3. The functionality of the application layer software ensures that all SCCP signaling and media packets can traverse the FWSM by providing NAT of the SCCP Signaling packets.

There are 5 versions of the SCCP protocol: 2.4, 3.0.4, 3.1.1, 3.2, and 3.3.2. The FWSM supports all versions through Version 3.3.2. The FWSM provides both PAT and NAT support for SCCP. PAT is necessary if you have limited numbers of global IP addresses for use by IP phones.

Normal traffic between the Cisco CallManager and Cisco IP Phones uses SCCP and is handled by SCCP inspection without any special configuration.The FWSM also supports DHCP options 150 and 66, which allow the FWSM to send the location of a TFTP server to Cisco IP Phones and other DHCP clients. For more information, see the **dhcp-server** command.

Supporting Cisco IP Phones

In topologies where Cisco CallManager is located on the higher security interface with respect to the Cisco IP Phones, if NAT is required for the Cisco CallManager IP address, the mapping must be **static** as a Cisco IP Phone requires the Cisco CallManager IP address to be specified explicitly in its configuration. An identity static entry allows the Cisco CallManager on the higher security interface to accept registrations from the Cisco IP Phones.

Cisco IP Phones require access to a TFTP server to download the configuration information they need to connect to the Cisco CallManager server.

When the Cisco IP Phones are on a lower security interface compared to the TFTP server, you must use an access list to connect to the protected TFTP server on UDP port 69. While you do need a static entry for the TFTP server, this does not have to be an "identity" static entry. When using NAT, an identity static entry maps to the same IP address. When using PAT, it maps to the same IP address and port.

When the Cisco IP Phones are on a *higher* security interface compared to the TFTP server and Cisco CallManager, no access list or static entry is required to allow the Cisco IP Phones to initiate the connection.

Restrictions and Limitations

The following are limitations that apply to the current version of PAT and NAT support for SCCP:

- PAT will not work with configurations using the **alias** command.
- Outside NAT or PAT is **not** supported.

Note Stateful Failover of SCCP calls is now supported except for calls that are in the middle of call setup.

If the address of an internal Cisco CallManager is configured for NAT or PAT to a different IP address or port, registrations for external Cisco IP Phones will fail because the FWSM currently does not support NAT or PAT for the file content transferred via TFTP. Although the FWSM does support NAT of TFTP messages, and opens a pinhole for the TFTP file to traverse the FWSM, the FWSM cannot translate the Cisco CallManager IP address and port embedded in the Cisco IP Phone configuration files that are being transferred using TFTP during phone registration.

Inspecting Signaling Messages

For inspecting signaling messages, the **inspect skinny** command often needs to determine locations of the media endpoints (for example, IP phones).

This information is used to prepare access-control and NAT state for media traffic to traverse the firewall transparently without manual configuration.

In determining these locations, the **inspect skinny** command does **not** use the tunnel default gateway route. A tunnel default gateway route is a route of the form **route** *interface* **0 0** *metric* **tunneled**. This route overrides the default route for packets that egress from IPSec tunnels. Therefore, if the **inspect skinny** command is desired for VPN traffic, do not configure the tunnel default gateway route. Instead, us other static routing or dynamic routing.

Examples

The following example shows how to enable the SCCP inspection engine, which creates a class map to match SCCP traffic on the default port (2000). The service policy is then applied to the outside interface.

```
hostname(config)# class-map skinny-port
hostname(config-cmap)# match port tcp eq 2000
hostname(config-cmap)# exit
hostname(config)# policy-map skinny_policy
```

```
hostname(config-pmap)# class skinny-port
hostname(config-pmap-c)# inspect skinny
hostname(config-pmap-c)# exit
hostname(config)# service-policy skinny_policy interface outside
```

To enable SCCP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug skinny	Enables SCCP debug information.
show skinny	Displays information about SCCP sessions established through the FWSM.
show conn	Displays the connection state for different connection types.
timeout	Sets the maximum idle time duration for different protocols and session types.

inspect smtp

To enable non-extended SMTP application inspection, use the **inspect smtp** command in class configuration mode. The class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect smtp

no inspect smtp

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol smtp** command, which is now deprecated.

Usage Guidelines

SMTP application inspection provides basic protection against SMTP-based attacks by restricting the types of SMTP commands that can pass through the FWSM and by adding monitoring capabilities. The application inspection process for SMTP does not include for extended SMTP sessions.

SMTP application inspection, as enabled by the **inspect smtp** command, occurs in fast path processing; therefore, it occurs on one of the three network processors on the FWSM.

The **inspect smtp** command includes the functionality previously provided by the **fixup smtp** command. It supports seven RFC 821 commands (DATA, HELO, MAIL, NOOP, QUIT, RCPT, RSET). Other SMTP and extended SMTP commands are not supported. Unsupported commands are translated into Xs, which are rejected by the internal server. This results in a message such as “500 Command unknown: 'XXX'.” Incomplete commands are discarded.

Note If a policy map contains both the **inspect smtp** command and the **inspect esmtp** command, only the first command listed in the policy map is applied to matching traffic.

The **inspect smtp** command changes the characters in the server SMTP banner to asterisks except for the “2”, “0”, “0” characters. Carriage return (CR) and linefeed (LF) characters are ignored.

With SMTP inspection enabled, a Telnet session used for interactive SMTP may hang if the following rules are not observed: SMTP commands must be at least four characters in length; must be terminated with carriage return and line feed; and must wait for a response before issuing the next reply.

An SMTP server responds to client requests with numeric reply codes and optional human readable strings. SMTP application inspection controls and reduces the commands that the user can use as well as the messages that the server returns. SMTP inspection performs three primary tasks:

- Restricts SMTP requests to seven basic SMTP commands.
- Monitors the SMTP command-response sequence.
- Generates an audit trail—Audit record 108002 is generated when invalid character embedded in the mail address is replaced. For more information, see RFC 821.

SMTP inspection monitors the command and response sequence for the following anomalous signatures:

- Truncated commands.
- Incorrect command termination (not terminated with <CR><LR>).
- The MAIL and RCPT commands specify who are the sender and the receiver of the mail. Mail addresses are scanned for strange characters. The pipeline character | is deleted (changed to a blank space) and | are only allowed if they are used to define a mail address | must be preceded by "<").
- Unexpected transition by the SMTP server.
- For unknown commands, the FWSM changes all the characters in the packet to X. In this case, the server will generate an error code to the client. Because of the change in the packet, the TCP checksum has to be recalculated or adjusted.
- TCP stream editing.
- Command pipelining.

Note FWSM supports SMTP and Extended SMTP Inspection for inbound traffic only; namely, FWSM supports inspection of traffic from a lower security level to a higher security level.

Examples

The following example shows how to enable the SMTP inspection engine, which creates a class map to match SMTP traffic on the default port (25). The service policy is then applied to the outside interface.

```
hostname(config)# class-map smtp-port
hostname(config-cmap)# match port tcp eq 25
hostname(config-cmap)# exit
hostname(config)# policy-map smtp_policy
hostname(config-pmap)# class smtp-port
hostname(config-pmap-c)# inspect smtp
hostname(config-pmap-c)# exit
hostname(config)# service-policy smtp_policy interface outside
```

To enable SMTP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug smtp	Enables debug information for SMTP.
inspect esmtp	Enables extended SMTP application inspection.

Commands	Description
policy-map	Associates a class map with specific security actions.
show conn	Displays the connection state for different connection types, including SMTP.

inspect snmp

To enable SNMP application inspection or to change the ports to which the FWSM listens, use the **inspect snmp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect snmp *map_name*

no inspect snmp *map_name*

Syntax Description

map_name	The name of the SNMP map.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

Use the **inspect snmp** command to enable SNMP inspection, using the settings configured with an SNMP map, which you create using the **snmp-map** command. Use the **deny version** command in SNMP map configuration mode to restrict SNMP traffic to a specific version of SNMP.

Earlier versions of SNMP are less secure so restricting SNMP traffic to Version 2 may be required by your security policy. To deny a specific version of SNMP, use the **deny version** command within an SNMP map, which you create using the **snmp-map** command. After configuring the SNMP map, you enable the map using the **inspect snmp** command and then apply it to one or more interfaces using the **service-policy** command.

Examples

The following example identifies SNMP traffic, defines an SNMP map, defines a policy, enables SNMP inspection, and applies the policy to the outside interface:

```
hostname(config)# access-list snmp-acl permit tcp any any eq 161
hostname(config)# access-list snmp-acl permit tcp any any eq 162
hostname(config)# class-map snmp-port
hostname(config-cmap)# match access-list snmp-acl
hostname(config-cmap)# exit
hostname(config)# snmp-map inbound_snmp
hostname(config-snmp-map)# deny version 1
```

```
hostname(config-snmp-map)# exit
hostname(config)# policy-map inbound_policy
hostname(config-pmap)# class snmp-port
hostname(config-pmap-c)# inspect snmp inbound_snmp
hostname(config-pmap-c)# exit
```

To enable strict snmp application inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
deny version	Disallows traffic using a specific version of SNMP.
snmp-map	Defines an SNMP map and enables SNMP map configuration mode.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect sqlnet

To enable Oracle SQL*Net application inspection, use the **inspect sqlnet** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect sqlnet

no inspect sqlnet

Syntax Description

This command has no arguments or keywords.

Defaults

This command is enabled by default.

The default port assignment is 1521.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol sqlnet** command, which is now deprecated.

Usage Guidelines

The SQL*Net protocol consists of different packet types that the FWSM handles to make the data stream appear consistent to the Oracle applications on either side of the FWSM.

The default port assignment for SQL*Net is 1521. This is the value used by Oracle for SQL*Net, but this value does not agree with IANA port assignments for Structured Query Language (SQL). Use the **class-map** command to apply SQL*Net inspection to a range of port numbers.

The FWSM NATs all addresses and looks in the packets for all embedded ports to open for SQL*Net Version 1.

For SQL*Net Version 2, all DATA or REDIRECT packets that immediately follow REDIRECT packets with a zero data length will be fixed up.

The packets that need fix-up contain embedded host/port addresses in the following format:

```
(ADDRESS=(PROTOCOL=tcp)(DEV=6)(HOST=a.b.c.d)(PORT=a))
```

Note The embedded, fully qualified domain name (FQDN) for the SQL server is not supported in an inspected SQL*Net redirect packet instead of the server IP address. Therefore, in the above the string, the text after "HOST=" must be in the form of an dotted-decimal IP address and cannot contain the FQDN of the SQL server.

SQL*Net Version 2 TNSFrame types (Connect, Accept, Refuse, Resend, and Marker) will not be scanned for addresses to NAT nor will inspection open dynamic connections for any embedded ports in the packet.

SQL*Net Version 2 TNSFrames, Redirect, and Data packets will be scanned for ports to open and addresses to NAT, if preceded by a REDIRECT TNSFrame type with a zero data length for the payload. When the Redirect message with data length zero passes through the FWSM, a flag will be set in the connection data Structure to expect the Data or Redirect message that follows to be NATed and ports to be dynamically opened. If one of the TNS frames in the preceding paragraph arrive after the Redirect message, the flag will be reset.

The SQL*Net inspection engine will recalculate the checksum, change IP, TCP lengths, and readjust Sequence Numbers and Acknowledgment Numbers using the delta of the length of the new and old message.

SQL*Net Version 1 is assumed for all other cases. TNSFrame types (Connect, Accept, Refuse, Resend, Marker, Redirect, and Data) and all packets will be scanned for ports and addresses. Addresses will be NATed and port connections will be opened.

Examples

The following example shows how to enable the SQL*Net inspection engine, which creates a class map to match SQL*Net traffic on the default port (1521). The service policy is then applied to the outside interface.

```
hostname(config)# class-map sqlnet-port
hostname(config-cmap)# match port tcp eq 1521
hostname(config-cmap)# exit
hostname(config)# policy-map sqlnet_policy
hostname(config-pmap)# class sqlnet-port
hostname(config-pmap-c)# inspect sqlnet
hostname(config-pmap-c)# exit
hostname(config)# service-policy sqlnet_policy interface outside
```

To enable SQL*Net inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug sqlnet	Enables debug information for SQL*Net.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.
show conn	Displays the connection state for different connection types, including SQL*net.

inspect sunrpc

To enable Sun RPC application inspection or to change the ports to which the FWSM listens, use the **inspect sunrpc** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect sunrpc

no inspect sunrpc

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced, replacing the **fixup protocol rpc** command, which is now deprecated.

Usage Guidelines

To enable Sun RPC application inspection or to change the ports to which the FWSM listens, use the **inspect sunrpc** command in policy map class configuration mode, which is accessible by using the **class** command within policy map configuration mode. To remove the configuration, use the **no** form of this command.

The **inspect sunrpc** command enables or disables application inspection for the Sun RPC protocol. Sun RPC is used by NFS and NIS. Sun RPC services can run on any port on the system. When a client attempts to access an Sun RPC service on a server, it must find out which port that service is running on. It does this by querying the portmapper process on the well-known port of 111.

The client sends the Sun RPC program number of the service, and gets back the port number. From this point on, the client program sends its Sun RPC queries to that new port. When a server sends out a reply, the FWSM intercepts this packet and opens both embryonic TCP and UDP connections on that port.

Note Sun RPC Inspection is not supported with the XLATE BYPASS feature because Sun RPC Inspection relies on XLATES for functionality and the XLATE BYPASS feature can prevent Sun RPC Inspection from functioning correctly. See the **xlate-bypass** command for more information about this feature.

Note NAT or PAT of Sun RPC payload information is not supported.

Examples

The following example shows how to enable the RPC inspection engine, which creates a class map to match RPC traffic on the default port (111). The service policy is then applied to the outside interface.

```
hostname(config)# class-map sunrpc-port
hostname(config-cmap)# match port tcp eq 111
hostname(config-cmap)# exit
hostname(config)# policy-map sample_policy
hostname(config-pmap)# class sunrpc-port
hostname(config-pmap-c)# inspect sunrpc
hostname(config-pmap-c)# exit
hostname(config)# service-policy sample_policy interface outside
```

To enable RPC inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
clear configure sunrpc_server	Removes the configuration performed using the **sunrpc-server** command.
clear sunrpc-server active	Clears the pinholes that are opened by Sun RPC application inspection for specific services, such as NFS or NIS.
show running-config sunrpc-server	Displays the information about the Sun RPC service table configuration.
sunrpc-server	Allows pinholes to be created with a specified timeout for Sun RPC services, such as NFS or NIS.
show sunrpc-server active	Displays the pinholes open for Sun RPC services.

inspect tftp

To disable TFTP application inspection, or to enable it if it has been previously disabled, use the **inspect tftp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect tftp

no inspect tftp

Syntax Description

This command has no arguments or keywords.

Defaults

This command is enabled by default.

The default port assignment is 69.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

Trivial File Transfer Protocol (TFTP), described in RFC 1350, is a simple protocol to read and write files between a TFTP server and client.

The FWSM inspects TFTP traffic and dynamically creates connections and translations, if necessary, to permit file transfer between a TFTP client and server. Specifically, the inspection engine inspects TFTP read request (RRQ), write request (WRQ), and error notification (ERROR).

A dynamic secondary channel and a PAT translation, if necessary, are allocated on a reception of a valid read (RRQ) or write (WRQ) request. This secondary channel is subsequently used by TFTP for file transfer or error notification.

Only the TFTP server can initiate traffic over the secondary channel, and at most one incomplete secondary channel can exist between the TFTP client and server. An error notification from the server closes the secondary channel.

TFTP inspection must be enabled if static PAT is used to redirect TFTP traffic.

Examples

The following example shows how to enable the TFTP inspection engine, which creates a class map to match TFTP traffic on the default port (69). The service policy is then applied to the outside interface.

```
hostname(config)# class-map tftp-port
hostname(config-cmap)# match port udp eq 69
hostname(config-cmap)# exit
hostname(config)# policy-map tftp_policy
hostname(config-pmap)# class tftp-port
hostname(config-pmap-c)# inspect tftp
hostname(config-pmap-c)# exit
hostname(config)# service-policy tftp_policy interface outside
```

To enable TFTP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect waas

To enable WAAS application inspection, use the **inspect waas** command in class configuration mode. The class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect waas

no inspect waas

Syntax Description

This command has no arguments or keywords.

Defaults

No default behaviors or values.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.2(1)	This command was introduced.

Examples

The following example shows how to enable WAAS application inspection:

```
hostname(config-pmap-c)# inspect waas
```

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.

inspect xdmcp

To enable XDMCP application inspection or to change the ports to which the FWSM listens, use the **inspect xdmcp** command in class configuration mode. Class configuration mode is accessible from policy map configuration mode. To remove the configuration, use the **no** form of this command.

inspect xdmcp

no inspect xdmcp

Syntax Description

This command has no arguments or keywords.

Defaults

This command is disabled by default.

Command Modes

The following table shows the modes in which you can enter the command:

	Firewall Mode		Security Context		
				Multiple	
Command Mode	Routed	Transparent	Single	Context	System
Class configuration	•	•	•	•	—

Command History

Release	Modification
3.1(1)	This command was introduced.

Usage Guidelines

The **inspect xdmcp** command enables or disables application inspection for the XDMCP protocol.

XDMCP is a protocol that uses UDP port 177 to negotiate X sessions, which use TCP when established.

For successful negotiation and start of an XWindows session, the FWSM must allow the TCP back connection from the Xhosted computer. To permit the back connection, use the **established** command on the FWSM. Once XDMCP negotiates the port to send the display, The **established** command is consulted to verify if this back connection should be permitted.

During the XWindows session, the manager talks to the display Xserver on the well-known port 6000 | n. Each display has a separate connection to the Xserver, as a result of the following terminal setting.

```
setenv DISPLAY Xserver:n
```

where *n* is the display number.

When XDMCP is used, the display is negotiated using IP addresses, which the FWSM can NAT if needed. XDCMP inspection does not support PAT.

Examples

The following example shows how to enable the XDMCP inspection engine, which creates a class map to match XDMCP traffic on the default port (177). The service policy is then applied to the outside interface.

```
hostname(config)# class-map xdmcp-port
hostname(config-cmap)# match port tcp eq 177
hostname(config-cmap)# exit
hostname(config)# policy-map xdmcp_policy
hostname(config-pmap)# class xdmcp-port
hostname(config-pmap-c)# inspect xdmcp
hostname(config-pmap-c)# exit
hostname(config)# service-policy xdmcp_policy interface outside
```

To enable XDMCP inspection for all interfaces, use the **global** parameter in place of **interface outside**.

Related Commands

Commands	Description
class-map	Defines the traffic class to which to apply security actions.
debug xdmcp	Enables debug information for XDMCP.
policy-map	Associates a class map with specific security actions.
service-policy	Applies a policy map to one or more interfaces.