



E-mail Inspection Engine

The E-mail Inspection Engine feature allows the Cisco IOS Firewall to inspect Post Office Protocol 3 (POP3) and Internet Message Access Protocol (IMAP) e-mail, in addition to Simple Mail Transfer Protocol (SMTP) and Extended Simple Mail Transfer Protocol (ESMTP) e-mail which were previously supported.

The **secure-login** enhancement allows people to download external POP3 e-mail only if authentication methods are secure.

Feature History for E-mail Inspection Engine

Release	Modification
12.3(14)T	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at <http://www.cisco.com/go/fn>. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

- [Prerequisites for E-mail Inspection Engine, page 2](#)
- [Restrictions for E-mail Inspection Engine, page 2](#)
- [Information About E-mail Inspection Engine, page 2](#)
- [How to Configure E-mail Inspection Engine, page 4](#)
- [Configuration Examples for E-mail Inspection Engine, page 7](#)
- [Additional References, page 8](#)



Corporate Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Copyright © 2005 Cisco Systems, Inc. All rights reserved.

- [Command Reference, page 9](#)
- [Glossary, page 31](#)

Prerequisites for E-mail Inspection Engine

- Configure CBAC.
- Enable SSL VPN tunnels.

Restrictions for E-mail Inspection Engine

None.

Information About E-mail Inspection Engine

To configure E-mail Inspection Engine, you need to understand the following concepts:

- [E-mail Inspection Engine Operation, page 2](#)
- [Inspection, page 3](#)
- [POP3, page 3](#)
- [IMAP Protocol, page 3](#)
- [Client Command Validation, page 4](#)
- [SMTP, page 4](#)
- [SSL, page 4](#)

E-mail Inspection Engine Operation

The client/server communication is validated from the time the TCP connection is initialized until the client is authenticated. The Cisco IOS Firewall uses a state router to track each stage of authentication. After the client is authenticated, the Cisco IOS Firewall allows all the client/server commands without further L7 inspection. TCP L4 inspection continues until the connection is closed. At the end of the e-mail session when the client host quits and before the TCP connection is closed, no further client/server interaction is allowed unless the client is reauthenticated.

During the authentication, any unrecognized command causes the Cisco IOS Firewall to drop the packet and close the connection.

If encryption is negotiated between the client and server control channel, no further validation occurs.

An e-mail client logging in from a nonsecure location may need to use encryption for authentication. For information about secure logins, see the description of the **secure-login** keyword of the **ip inspect name** command.

Inspection

Context Based Access Control (CBAC) inspects traffic that travels through the firewall to discover and manage state information for TCP and User Datagram Protocol (UDP) sessions. This state information is used to create temporary openings in the firewall's access lists to allow return traffic and additional data connections for permissible sessions.

Inspecting packets at the application layer and maintaining TCP and UDP session information provides CBAC with the ability to detect and prevent certain types of network attacks such as SYN-flooding. A SYN-flood attack occurs when a network attacker floods a server with a barrage of requests for connection and does not complete the connection. The resulting volume of half-open connections can overwhelm the server, causing it to deny service to valid requests. Network attacks that deny access to a network device are called denial-of-service (DoS) attacks.

POP3

The Post Office Protocol, Version 3 (POP3) is used to receive e-mail that is stored on a mail server. Unlike IMAP, POP only retrieves mail from a remote host.

POP3 works best when there is only one computer because it supports "offline" message access where messages are downloaded and then deleted from the mail server. This mode of access is not compatible with access from multiple computers because it tends to sprinkle messages across all the computers used for mail access.

With POP3-based e-mail clients, messages are downloaded to the user's local message store and can also be deleted from the mail server. Deletion is optional in most clients. When a new voice message arrives, the subscriber's only immediate notification is the activation of the MWI on the phone. New messages are displayed in the Inbox only after the client's local message store is updated with the Exchange message store. After the subscriber downloads new messages, the message state automatically changes from "new" to "read" on the server, even though the subscriber has not actually listened to the voice messages. MWIs on the subscriber's phone are extinguished, and the message state between the TUI and the subscriber's Inbox are not synchronized.

IMAP Protocol

The Internet Message Access Protocol (IMAP) is a method of accessing electronic mail or bulletin board messages that are kept on a mail server that may be shared. It permits a "client" e-mail program to access remote messages as though they were local. For example, e-mail stored on an IMAP server can be retrieved, sent, and managed from a desktop computer at home, from a workstation at the office, or from a laptop without transferring messages or files back and forth between the computers.

Only the message header and sender information are displayed in the Inbox until the user downloads the entire message, including attachments, from the server. When a new voice message arrives, the subscriber's only immediate notification is the activation of the Message Waiting Indication (MWI) on the phone. New messages are displayed in the Inbox only after the client's local message store is updated with the Exchange message store. When the subscriber listens to a new message by using the telephone user interface (TUI), the MWI is extinguished. In this case again, the message state is not updated in the Inbox until the client's message store is refreshed. However, if the subscriber uses an installed multimedia player to listen to the WaveForm Audio (WAV) attachment from the e-mail client's Inbox, message state changes are automatically synchronized with the TUI.

How message state changes are conveyed to the Cisco Unity subscriber, and how these changes are synchronized with the TUI, depend on whether the subscriber's e-mail client is configured to use POP3 or IMAP4 to access Exchange.

Client Command Validation

The Cisco IOS Firewall authenticates an e-mail client accessing an IMAP or POP3 server before allowing complete access into the server. The firewall searches the IMAP/POP3 TCP stream for valid protocol commands. If the client's commands are outside the protocol's definition, the Cisco IOS Firewall drops the packets and resets the connection.

Client command validation is typically needed in a DeMilitarized Zone (DMZ). Client access is allowed into the DMZ only if the e-mail server validates the user authentication. After the client is authenticated, the client becomes a trusted user and access is permitted.

SMTP

The Simple Mail Transfer Protocol (SMTP) is used to transfer e-mail between servers and clients on the Internet. E-mail clients and mail servers that use protocols other than Message Application Programming Interface (MAPI) can use the SMTP protocol to transfer a message from a client to the server, and then forward it to a message recipient's server. To retrieve, send, and manage these messages from the e-mail client use POP3 or IMAP4.

Cisco Unity uses SMTP to route voice messages via the Internet Voice Connector (IVC) gateway between other Exchange servers that are not connected by using a Site Message Connector. There is an IVC gateway on either end of the SMTP connection between Exchange servers. This ensures that MAPI message attributes survive the outbound transit between SMTP connections. It also ensures that the MIME-encoded attributes survive the inbound transit, and are included with the message stored in the Exchange message store.

SSL

The Secure Socket Layer (SSL) protocol is the standard protocol that delivers secure content over the Internet. It is a point-to-point security protocol that secures communication between a client and a server. SSL usually does not require a special client (that is, a Web browser often will suffice) and it does not require any additional operating system software.

SSL includes client and server authentication and data encryption for a limited set of applications (for example, the Web, e-mail, news, and file transfer). SSL is useful for securing e-commerce transactions over the Internet, and the protocol is well suited for extranets and remote access because it is relatively simple to deploy.

How to Configure E-mail Inspection Engine

This section contains the following procedures:

- [Configuring Firewall Inspection of POP3 or IMAP E-mail, page 5](#) (required)
- [Verifying the E-mail Inspection Engine Configuration, page 6](#) (optional)

Configuring Firewall Inspection of POP3 or IMAP E-mail

To allow the Cisco IOS Firewall to inspect POP3 or IMAP e-mail, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-name protocol* [**alert {on | off}**] [**audit-trail {on | off}**][**reset**]
[**secure-login**] [**timeout seconds**]
4. **interface** *type slot/port*
5. **ip inspect name** *inspection-name* {**in** | **out**}
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none">Enter your password if prompted.
Step 2	configure terminal Example: Router# configure terminal	Enters global configuration mode.
Step 3	ip inspect name inspection-name protocol [alert {on off}] [audit-trail {on off}][reset] [secure-login] [timeout seconds] Example: Router(config)# ip inspect name mail-guard pop3	Defines a set of inspection rules.
Step 4	interface type slot/port Example: Router(config-if)# interface 1/0	Configures an interface type.
Step 5	ip inspect name inspection-name {in out} Example: Router(config-if)# ip inspect name mail-guard in	Enables the Cisco IOS Firewall on an interface.
Step 6	exit Example: Router(config-if)# exit	Exits interface configuration mode and returns to global configuration mode.

Verifying the E-mail Inspection Engine Configuration

To verify the E-mail Inspection Engine configuration, perform the following steps.

SUMMARY STEPS

1. **debug ip inspect imap**
2. **debug ip inspect pop3**
3. **show ip inspect {name inspection-name | config | interfaces | session [detail] | all}**

DETAILED STEPS

Step 1 **debug ip inspect imap**

Use this command to display messages about Cisco IOS Firewall events related to IMAP protocol e-mail messages.

```
Router# debug ip inspect imap
```

Step 2 **debug ip inspect pop3**

Use this command to display messages about Cisco IOS Firewall events related to POP3 protocol e-mail messages.

```
Router# debug ip inspect pop3
```

Step 3 **show ip inspect {name *inspection-name* | config | interfaces | session [detail] | all}**

Use this command to view CBAC configuration and session information.

```
Router# show ip inspect
```

```
Session audit trail is disabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
  Inspection name mail-guard
    tcp timeout 3600
    udp timeout 30
    ftp timeout 3600
```

Configuration Examples for E-mail Inspection Engine

- [Configuring IMAP and POP3 Protocol E-mail: Example, page 7](#)

Configuring IMAP and POP3 Protocol E-mail: Example

The following example configures the Cisco IOS Firewall inspection of IMAP and POP3 protocol e-mail:

```
configure terminal
ip inspect name mail-guard pop3
ip inspect name mail-guard imap
exit
```

The following commands enable this functionality on an interface:

```
configure terminal
interface 1/0
ip inspect name mail-guard in
exit
```

Additional References

The following sections provide references related to E-Mail Inspection Engine.

Related Documents

Related Topic	Document Title
IMAP and POP3	White Paper: <i>Deploying Cisco Unity in Diverse Messaging Environments (All Versions with Microsoft Exchange)</i>
CBAC	<i>Cisco IOS Security Configuration Guide</i> , Release 12.3 <i>Cisco IOS Security Command Reference</i> , Release 12.3T

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 1939	J Myers and M. Rose, “Post Office Protocol, Version 3 (POP3),” May 1996.
RFC 3501	M. Crispin, “ <i>Internet Message Access Protocol (IMAP4rev1)</i> ,” March 2003.

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

This section documents new and modified commands. All other commands used with this feature are documented in the Cisco IOS Release 12.3 command reference publications.

- [debug ip inspect](#)
- [ip inspect name](#)
- [show ip inspect](#)

debug ip inspect

To display messages about Cisco IOS Firewall events, use the **debug ip inspect** command in privileged EXEC mode. To disable debugging output, use the **no** form of this command.

debug ip inspect { **function-trace** | **object-creation** | **object-deletion** | **events** | **timers** | *protocol* | **detailed** }

no debug ip inspect

Syntax Description

function-trace	Displays messages about software functions called by the Cisco IOS Firewall.
object-creation	Displays messages about software objects being created by the Cisco IOS Firewall. Object creation corresponds to the beginning of Cisco IOS Firewall-inspected sessions.
object-deletion	Displays messages about software objects being deleted by the Cisco IOS Firewall. Object deletion corresponds to the closing of Cisco IOS Firewall-inspected sessions.
events	Displays messages about Cisco IOS Firewall software events, including information about Cisco IOS Firewall packet processing.
timers	Displays messages about Cisco IOS Firewall timer events such as when the Cisco IOS Firewall idle timeout is reached.
<i>protocol</i>	Displays messages about Cisco IOS Firewall-inspected protocol events, including details about the packets of the protocol. Table 1 provides a list of <i>protocol</i> keywords.
detailed	Displays detailed information to be displayed for all the other enabled Cisco IOS Firewall debugging. Use this form of the command in conjunction with other Cisco IOS Firewall debug commands.

Table 1 Protocol Keywords for the debug ip inspect Command

Application Protocol	Protocol Keyword
Transport-layer protocols	
ICMP	icmp
TCP	tcp
User Datagram Protocol (UDP)	udp
Application-layer protocols	
CU-SeeMe	cuseeme
FTP commands and responses	ftp-cmd
FTP tokens (enables tracing of the FTP tokens parsed)	ftp-tokens
H.323 (version 1 and version 2)	h323
HTTP	http
IMAP	imap
Microsoft NetShow	netshow

Table 1 Protocol Keywords for the debug ip inspect Command (Continued)

Application Protocol	Protocol Keyword
POP3	pop3
RealAudio	realaudio
Remote procedure call (RPC)	rpc
Real Time Streaming Protocol (RTSP)	rtsp
Session Initiation Protocol (SIP)	sip
Simple Mail Transfer Protocol (SMTP)	smtp
Skinny Client Control Protocol (SCCP)	skinny
Structured Query Language*Net (SQL*Net)	sqlnet
StreamWorks	streamworks
TFTP	tftp
UNIX r-commands (rlogin, rexec, rsh)	rcmd
VDOLive	vdolive

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2 P	This command was introduced.
12.0(5)T	NetShow support was added.
12.0(7)T	H.323 V2 and RTSP protocol support were added.
12.2(11)YU	Support for the ICMP and SIP protocols was added.
12.2(15)T	This command was integrated into Cisco IOS Release 12.2(15)T.
12.3(1)	Support for the skinny protocol was added.
12.3(14)T	Support for the IMAP and POP3 protocols was added.

Examples

The following is sample output from the **debug ip inspect function-trace** command:

```
Router# debug ip inspect function-trace

*Mar  2 01:16:16: CBAC FUNC: insp_inspection
*Mar  2 01:16:16: CBAC FUNC: insp_pre_process_sync
*Mar  2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
*Mar  2 01:16:16: CBAC FUNC: insp_find_pregen_session
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_get_irc_of_idb
*Mar  2 01:16:16: CBAC FUNC: insp_get_idbsb
*Mar  2 01:16:16: CBAC FUNC: insp_create_sis
*Mar  2 01:16:16: CBAC FUNC: insp_inc_halfopen_sis
*Mar  2 01:16:16: CBAC FUNC: insp_link_session_to_hash_table
*Mar  2 01:16:16: CBAC FUNC: insp_inspect_pak
*Mar  2 01:16:16: CBAC FUNC: insp_l4_inspection
*Mar  2 01:16:16: CBAC FUNC: insp_process_tcp_seg
```

```

*Mar 2 01:16:16: CBAC FUNC: insp_listen_state
*Mar 2 01:16:16: CBAC FUNC: insp_ensure_return_traffic
*Mar 2 01:16:16: CBAC FUNC: insp_add_acl_item
*Mar 2 01:16:16: CBAC FUNC: insp_ensure_return_traffic
*Mar 2 01:16:16: CBAC FUNC: insp_add_acl_item
*Mar 2 01:16:16: CBAC FUNC: insp_process_syn_packet
*Mar 2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41
*Mar 2 01:16:16: CBAC FUNC: insp_create_tcp_host_entry
*Mar 2 01:16:16: CBAC* FUNC: insp_fast_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_inspect_pak
*Mar 2 01:16:16: CBAC* FUNC: insp_l4_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_process_tcp_seg
*Mar 2 01:16:16: CBAC* FUNC: insp_synrcvd_state
*Mar 2 01:16:16: CBAC* FUNC: insp_fast_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_inspect_pak
*Mar 2 01:16:16: CBAC* FUNC: insp_l4_inspection
*Mar 2 01:16:16: CBAC* FUNC: insp_process_tcp_seg
*Mar 2 01:16:16: CBAC* FUNC: insp_synrcvd_state
*Mar 2 01:16:16: CBAC FUNC: insp_dec_halfopen_sis
*Mar 2 01:16:16: CBAC FUNC: insp_remove_sis_from_host_entry
*Mar 2 01:16:16: CBAC FUNC: insp_find_tcp_host_entry addr 40.0.0.1 bucket 41

```

This output shows the functions called by the Cisco IOS Firewall as a session is inspected. Entries with an asterisk (*) after the word “CBAC” are entries when the fast path is used; otherwise, the process path is used.

The following is sample output from the **debug ip inspect object-creation** and **debug ip inspect object-deletion** commands:

```

Router# debug ip inspect object-creation
Router# debug ip inspect object-deletion

*Mar 2 01:18:30: CBAC OBJ_CREATE: create pre-gen sis 25A3574
*Mar 2 01:18:30: CBAC OBJ_CREATE: create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:30: CBAC OBJ_CREATE: create sis 25C1CC4
*Mar 2 01:18:30: CBAC OBJ_DELETE: delete pre-gen sis 25A3574
*Mar 2 01:18:30: CBAC OBJ_CREATE: create host entry 25A3574 addr 10.0.0.1 bucket 31
*Mar 2 01:18:30: CBAC OBJ_DELETE: delete sis 25C1CC4
*Mar 2 01:18:30: CBAC OBJ_DELETE: delete create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:31: CBAC OBJ_DELETE: delete host entry 25A3574 addr 10.0.0.1

```

The following is sample output from the **debug ip inspect object-creation**, **debug ip inspect object-deletion**, and **debug ip inspect events** commands:

```

Router# debug ip inspect object-creation
Router# debug ip inspect object-deletion
Router# debug ip inspect events

*Mar 2 01:18:51: CBAC OBJ_CREATE: create pre-gen sis 25A3574
*Mar 2 01:18:51: CBAC OBJ_CREATE: create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:51: CBAC Src 10.1.0.1 Port [1:65535]
*Mar 2 01:18:51: CBAC Dst 10.0.0.1 Port [46406:46406]
*Mar 2 01:18:51: CBAC Pre-gen sis 25A3574 created: 10.1.0.1[1:65535]
30.0.0.1[46406:46406]
*Mar 2 01:18:51: CBAC OBJ_CREATE: create sis 25C1CC4
*Mar 2 01:18:51: CBAC sis 25C1CC4 initiator_addr (10.1.0.1:20) responder_addr
(30.0.0.1:46406) initiator_alt_addr (40.0.0.1:20) responder_alt_addr (10.0.0.1:46406)
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete pre-gen sis 25A3574
*Mar 2 01:18:51: CBAC OBJ_CREATE: create host entry 25A3574 addr 10.0.0.1 bucket 31
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete sis 25C1CC4
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete create acl wrapper 25A36FC -- acl item 25A3634
*Mar 2 01:18:51: CBAC OBJ_DELETE: delete host entry 25A3574 addr 10.0.0.1

```

The following is sample output from the **debug ip inspect timers** command:

```
Router# debug ip inspect timers

*Mar 2 01:19:15: CBAC Timer Init Leaf: Pre-gen sis 25A3574
*Mar 2 01:19:15: CBAC Timer Start: Pre-gen sis 25A3574 Timer: 25A35D8 Time: 30000
milisecs
*Mar 2 01:19:15: CBAC Timer Init Leaf: sis 25C1CC4
*Mar 2 01:19:15: CBAC Timer Stop: Pre-gen sis 25A3574 Timer: 25A35D8
*Mar 2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 30000 milisecs
*Mar 2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 3600000 milisecs
*Mar 2 01:19:15: CBAC Timer Start: sis 25C1CC4 Timer: 25C1D5C Time: 5000 milisecs
*Mar 2 01:19:15: CBAC Timer Stop: sis 25C1CC4 Timer: 25C1D5C
```

The following is sample output from the **debug ip inspect tcp** command:

```
Router# debug ip inspect tcp

*Mar 2 01:20:43: CBAC* sis 25A3604 pak 2541C58 TCP P ack 4223720032 seq 4200176225(22)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar 2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar 2 01:20:43: CBAC sis 25A3604 pak 2541C58 TCP P ack 4223720032 seq 4200176225(22)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar 2 01:20:43: CBAC sis 25A3604 ftp L7 inspect result: PASS packet
*Mar 2 01:20:43: CBAC* sis 25A3604 pak 2544374 TCP P ack 4200176247 seq 4223720032(30)
(10.0.0.1:46409) <= (10.1.0.1:21)
*Mar 2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PASS packet
*Mar 2 01:20:43: CBAC* sis 25A3604 pak 25412F8 TCP P ack 4223720062 seq 4200176247(15)
(10.0.0.1:46409) => (10.1.0.1:21)
*Mar 2 01:20:43: CBAC* sis 25A3604 ftp L7 inspect result: PASS packet
*Mar 2 01:20:43: CBAC sis 25C1CC4 pak 2544734 TCP S seq 4226992037(0) (10.1.0.1:20) =>
(10.0.0.1:46411)
*Mar 2 01:20:43: CBAC* sis 25C1CC4 pak 2541E38 TCP S ack 4226992038 seq 4203405054(0)
(10.1.0.1:20) <= (10.0.0.1:46411)
```

This sample shows TCP packets being processed and lists the corresponding acknowledge (ACK) packet numbers and sequence (SEQ) numbers. The number of data bytes in the TCP packet is shown in parentheses—for example, (22). For each packet shown, the addresses and port numbers are shown separated by a colon. For example, (10.1.0.1:21) indicates an IP address of 10.1.0.1 and a TCP port number of 21.

Entries with an asterisk (*) after the word “CBAC” are entries when the fast path is used; otherwise, the process path is used.

The following is sample output from the **debug ip inspect tcp** and **debug ip inspect detailed** commands:

```
Router# debug ip inspect tcp
Router# debug ip inspect detailed

*Mar 2 01:20:58: CBAC* Pak 2541E38 Find session for (30.0.0.1:46409) (40.0.0.1:21) tcp
*Mar 2 01:20:58: P ack 4223720160 seq 4200176262(22)
*Mar 2 01:20:58: CBAC* Pak 2541E38 Addr:port pairs to match: (30.0.0.1:46409)
(40.0.0.1:21)
*Mar 2 01:20:58: CBAC* sis 25A3604 SIS_OPEN
*Mar 2 01:20:58: CBAC* Pak 2541E38 IP: s=30.0.0.1 (Ethernet0), d=40.0.0.1 (Ethernet1),
len 76,proto=6
*Mar 2 01:20:58: CBAC sis 25A3604 Saving State: SIS_OPEN/ESTAB iisn 4200176160 i_rcvnxt
4223720160 i_sndnxt 4200176262 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
4223720160 r_rcvwnd 8760
*Mar 2 01:20:58: CBAC* sis 25A3604 pak 2541E38 TCP P ack 4223720160 seq 4200176262(22)
(30.0.0.1:46409) => (40.0.0.1:21)
*Mar 2 01:20:58: CBAC* sis 25A3604 pak 2541E38 SIS_OPEN/ESTAB TCP seq 4200176262(22)
Flags: ACK 4223720160 PSH
```

```

*Mar  2 01:20:58: CBAC* sis 25A3604 pak 2541E38 --> SIS_OPEN/ESTAB iisn 4200176160
i_rcvnxt 4223720160 i_sndnxt 4200176284 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262
r_sndnxt 4223720160 r_rcvwnd 8760
*Mar  2 01:20:58: CBAC* sis 25A3604 L4 inspect result: PASS packet 2541E38
(30.0.0.1:46409) (40.0.0.1:21) bytes 22 ftp
*Mar  2 01:20:58: CBAC sis 25A3604 Restoring State: SIS_OPEN/ESTAB iisn 4200176160
i_rcvnxt 4223
720160 i_sndnxt 4200176262 i_rcvwnd 8760 risn 4223719771 r_rcvnxt 4200176262 r_sndnxt
4223720160 r_rcvwnd 8760
*Mar  2 01:20:58: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar  2 01:20:58: CBAC* sis 25A3604 ftp L7 inspect result: PROCESS-SWITCH packet
*Mar  2 01:20:58: CBAC* Bump up: inspection requires the packet in the process
path(30.0.0.1) (40.0.0.1)
*Mar  2 01:20:58: CBAC Pak 2541E38 Find session for (30.0.0.1:46409) (40.0.0.1:21) tcp
*Mar  2 01:20:58: P ack 4223720160 seq 4200176262(22)
*Mar  2 01:20:58: CBAC Pak 2541E38 Addr:port pairs to match: (30.0.0.1:46409)
(40.0.0.1:21)
*Mar  2 01:20:58: CBAC sis 25A3604 SIS_OPEN
*Mar  2 01:20:58: CBAC Pak 2541E38 IP: s=30.0.0.1 (Ethernet0), d=40.0.0.1 (Ethernet1), len
76, proto=6

```

The following is sample output from the **debug ip inspect icmp** and **debug ip inspect detailed** commands:

```

Router# debug ip inspect icmp
Router# debug ip inspect detailed

```

```

1w6d:CBAC sis 81073F0C SIS_CLOSED
1w6d:CBAC Pak 80D2E9EC IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1
1w6d:CBAC ICMP:sis 81073F0C pak 80D2E9EC SIS_CLOSED ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC ICMP:start session from 192.168.133.3
1w6d:CBAC sis 81073F0C --> SIS_OPENING (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC sis 81073F0C L4 inspect result:PASS packet 80D2E9EC (192.168.133.3:0)
(0.0.0.0:0) bytes 56 icmp
1w6d:CBAC sis 81073F0C SIS_OPENING
1w6d:CBAC Pak 80E72BFC IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
1w6d:CBAC ICMP:sis 81073F0C pak 80E72BFC SIS_OPENING ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC sis 81073F0C L4 inspect result:PASS packet 80E72BFC (0.0.0.0:0)
(192.168.133.3:0) bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80D2F2C8 IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80D2F2C8 SIS_OPEN ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80D2F2C8 (192.168.133.3:0)
(0.0.0.0:0) bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80E737CC IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80E737CC SIS_OPEN ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80E737CC (0.0.0.0:0)
(192.168.133.3:0) bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80F554F0 IP:s=192.168.133.3 (Ethernet1), d=0.0.0.0 (Ethernet0), len 98,
proto=1

```

```
1w6d:CBAC* ICMP:sis 81073F0C pak 80F554F0 SIS_OPEN ICMP packet (192.168.133.3:0) =>
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80F554F0 (192.168.133.3:0)
(0.0.0.0:0) bytes 56 icmp
1w6d:CBAC* sis 81073F0C SIS_OPEN
1w6d:CBAC* Pak 80E73AC0 IP:s=0.0.0.0 (Ethernet0), d=192.168.133.3 (Ethernet1), len 98,
proto=1
1w6d:CBAC* ICMP:sis 81073F0C pak 80E73AC0 SIS_OPEN ICMP packet (192.168.133.3:0) <=
(0.0.0.0:0) datalen 56
1w6d:CBAC* sis 81073F0C --> SIS_OPEN (192.168.133.3:0) (0.0.0.0:0)
1w6d:CBAC* sis 81073F0C L4 inspect result:PASS packet 80E73AC0 (0.0.0.0:0) (192.168.133.3:0)
bytes 56 icmp
```

ip inspect name

To define a set of inspection rules, use the **ip inspect name** command in global configuration mode. To remove the inspection rule for a protocol or to remove the entire set of inspection rules, use the **no** form of this command.

```
ip inspect name inspection-name [parameter max-sessions number] protocol [alert {on | off}]
[audit-trail {on | off}] [timeout seconds]
```

```
no ip inspect name inspection-name [parameter max-sessions number] protocol [alert {on | off}]
[audit-trail {on | off}] [timeout seconds]
```

HTTP Inspection Syntax

```
ip inspect name inspection-name http [urlfilter] [java-list access-list] [alert {on | off}]
[audit-trail {on | off}] [timeout seconds]
```

```
no ip inspect name inspection-name protocol
```

SMTP and ESMTP Inspection Syntax

```
ip inspect name inspection-name {smtp | esmtplib} [alert {on | off}] [audit-trail {on | off}]
[max-data number] [timeout seconds]
```

remote-procedure call (RPC) Inspection Syntax

```
ip inspect name inspection-name [parameter max-sessions number] rpc program-number
number [wait-time minutes] [alert {on | off}] [audit-trail {on | off}] [timeout seconds]
```

```
no ip inspect name inspection-name protocol
```

POP3/IMAP Inspection Syntax

```
ip inspect name inspection-name imap [alert {on | off}] [audit-trail {on | off}] [reset]
[secure-login] [timeout number]
```

```
ip inspect name inspection-name pop3 [alert {on | off}] [audit-trail {on | off}] [reset]
[secure-login] [timeout number]
```

Fragment Inspection Syntax

```
ip inspect name inspection-name [parameter max-sessions number] fragment [max number
timeout seconds]
```

```
no ip inspect name inspection-name [parameter max-sessions number] fragment [max number
timeout seconds]
```

Application Firewall Provisioning Syntax

```
ip inspect name inspection-name [parameter max-sessions number] appfw policy-name
```

```
no ip inspect name inspection-name [parameter max-sessions number] appfw policy-name
```

User-Defined Application Syntax

ip inspect name *inspection-name* *user-10* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]

no ip inspect name *inspection-name* *user-10* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]

Session Limiting Syntax

no ip inspect name *inspection-name* [**parameter max-sessions** *number*]

Syntax Description

<i>inspection-name</i>	Names the set of inspection rules. If you want to add a protocol to an existing set of rules, use the same <i>inspection-name</i> as the existing set of rules. Note The <i>inspection-name</i> cannot exceed 16 characters; otherwise, the name will be truncated to the 16-character limit.
parameter max-sessions <i>number</i>	(Optional) Limits the number of established firewall sessions that a firewall rule creates. The default is that there is no limit to the number of firewall sessions.
<i>protocol</i>	A protocol keyword listed in Table 2 or Table 3 .
alert { on off }	(Optional) For each inspected protocol, the generation of alert messages can be set be on or off . If no option is selected, alerts are generated on the basis of the setting of the ip inspect alert-off command.
audit-trail { on off }	(Optional) For each inspected protocol, audit trail can be set on or off . If no option is selected, an audit trail message are generated on the basis of the setting of the ip inspect audit-trail command.
timeout <i>seconds</i>	(Optional) To override the global TCP or User Datagram Protocol (UDP), or Internet Control Message Protocol (ICMP) idle timeouts for the specified protocol, specify the number of seconds for a different idle timeout. This timeout overrides the global TCP, UDP, or ICMP timeouts but will not override the global Domain Name System (DNS) timeout.
http	Specifies the HTTP protocol for Java applet blocking.
urlfilter	(Optional) Associates URL filtering with HTTP inspection.
java-list <i>access-list</i>	(Optional) Specifies the numbered standard access list to use to determine “friendly” sites. This keyword is available only for the HTTP protocol, for Java applet blocking. Java blocking only works with numbered standard access lists.
smtp esmtplib	Specifies the protocol being used to inspect the traffic.
max-data <i>number</i>	(Optional) Specifies the maximum number of bytes (data) that can be transferred in a single Simple Mail Transport Protocol (SMTP) session. After the maximum value is exceeded, the firewall logs an alert message and closes the session. Default value: 20 MB
rpc program-number <i>number</i>	Specifies the program number to permit. This keyword is available only for the remote-procedure call protocol.

wait-time <i>minutes</i>	(Optional) Specifies the number of minutes to keep a small hole in the firewall to allow subsequent connections from the same source address and to the same destination address and port. The default wait-time is zero minutes. This keyword is available only for the remote-procedure call (RPC) protocol.
reset	(Optional) Resets the TCP connection if the client enters a non-protocol command before authentication is complete.
secure-login	(Optional) Causes a user at a non-secure location to use encryption for authentication.
imap	Specifies that the Internet Message Access Protocol (IMAP) is being used.
pop3	Specifies that the Post Office Protocol, Version 3 (POP3) is being used.
fragment	Specifies fragment inspection for the named rule.
max <i>number</i>	(Optional) Specifies the maximum number of unassembled packets for which state information (structures) is allocated by Cisco IOS software. Unassembled packets are packets that arrive at the router interface before the initial packet for a session. The acceptable range is 50 through 10000. The default is 256 state entries. Memory is allocated for the state structures, and setting this value to a larger number may cause memory resources to be exhausted.
timeout <i>seconds</i> (fragmentation)	(Optional) Configures the number of seconds that a packet state structure remains active. When the timeout value expires, the router drops the unassembled packet, freeing that structure for use by another packet. The default timeout value is 1 second. If this number is set to a value greater than 1 second, it is automatically adjusted by the Cisco IOS software when the number of free state structures goes below certain thresholds: when the number of free states is fewer than 32, the timeout is divided by 2. When the number of free states is fewer than 16, the timeout is set to 1 second.
appfw	Specifies application firewall provisioning.
<i>policy-name</i>	Application firewall policy name. Note This name must match the name specified via the appfw policy-name command.
<i>appname</i>	Specifies a user- or a system-defined application; for example, user-payroll-sap and user-sametime . Application names can contain hyphens and underscores; however, a user-defined application must have the prefix user- in its title.
port	Specifies the port range for an application.
tcp udp	Specifies the protocol being used to inspect the traffic.
from <i>begin_port_num to end_port_num port_num1 ...</i>	Specifies the starting and ending port numbers or a range of ports from 1 to 5. You must use the from and to keywords together.
list <i>acl_list_num</i>	(Optional) Specifies an access control list number. Only standard ACLs are supported.
description <i>description_string</i>	(Optional) Specifies a description of up to 40 characters.

<i>user-10</i>	Represents a user-defined application in the port-to-application mapping (PAM) table of the ip port-map command.
router-traffic	(Optional) Enables inspection of traffic destined to or originated from a router. Applicable only for H.323, TCP, and UDP protocols. For the command format, see the Note after Table 2 .

Defaults

No inspection rules are defined until you define them using this command.

no ip inspect-name *protocol* removes the inspection rule for the specified protocol.

no ip inspect name removes the entire set of inspection rules.

Command Modes

Global configuration

Command History

Release	Modification
11.2 P	This command was introduced.
12.0(5)T	Introduced configurable alert and audit trail, IP fragmentation checking, and NetShow protocol support.
12.2(11)YU	Support was added for ICMP and SIP protocols and the urlfilter keyword was added to the HTTP inspection syntax.
12.2(15)T	Support was added for ICMP, SIP protocols, and the urlfilter keyword was integrated into Cisco IOS Release 12.2(15)T.
12.3(1)	Skinny protocol support was added.
12.3(7)T	Extended Simple Mail Transfer Protocol (ESMTP) protocol support was added.
12.3(14)T	The appfw keyword and the <i>policy-name</i> argument were added to support application firewall provisioning. The parameter max-sessions, secure-login, reset, and router-traffic keywords were added. Support for a larger list of protocols including user-defined applications was added.

Usage Guidelines

To define a set of inspection rules, enter this command for each protocol that you want the Cisco IOS firewall to inspect, using the same *inspection-name*. Give each set of inspection rules a unique *inspection-name*, which should not exceed the 16-character limit. Define either one or two sets of rules per interface—you can define one set to examine both inbound and outbound traffic, or you can define two sets: one for outbound traffic and one for inbound traffic.

To define a single set of inspection rules, configure inspection for all the desired application-layer protocols, and for ICMP, TCP, and UDP, or as desired. This combination of TCP, UDP, and application-layer protocols join together to form a single set of inspection rules with a unique name. (There are no application-layer protocols associated with ICMP.)

To remove the inspection rule for a protocol, use the **no** form of this command with the specified inspection name and protocol; to remove the entire set of inspection rules, use the **no** form of this command only; that is, do not list any inspection names or protocols.

In general, when inspection is configured for a protocol, return traffic entering the internal network will be permitted only if the packets are part of a valid, existing session for which state information is being maintained.

Table 2 Protocol Keywords—Transport-Layer and Network-Layer Protocols

Protocol	Keyword
ICMP	icmp
TCP	tcp
UDP	udp

Note The TCP, UDP, and H.323 protocols support the **router-traffic** keyword, which enables inspection of traffic destined to or originated from a router. The command format is as follows:

```
ip inspect name inspection-name {TCP | UDP | H323} [alert {on | off}] [audit-trail {on | off}] [router-traffic] [timeout seconds]
```

TCP and UDP Inspection

You can configure TCP and UDP inspection to permit TCP and UDP packets to enter the internal network through the firewall, even if the application-layer protocol is not configured to be inspected. However, TCP and UDP inspection do not recognize application-specific commands, and therefore might not permit all return packets for an application, particularly if the return packets have a different port number from the previous exiting packet.

Any application-layer protocol that is inspected will take precedence over the TCP or UDP packet inspection. For example, if inspection is configured for FTP, all control channel information will be recorded in the state table, and all FTP traffic will be permitted back through the firewall if the control channel information is valid for the state of the FTP session. The fact that TCP inspection is configured is irrelevant.

With TCP and UDP inspection, packets entering the network must exactly match an existing session: the entering packets must have the same source or destination addresses and source or destination port numbers as the exiting packet (but reversed). Otherwise, the entering packets will be blocked at the interface.

Granular protocol inspection allows you to specify TCP or UDP ports by using the PAM table. This eliminates having to inspect all applications running under TCP or UDP and the need for multiple access control lists (ACLs) to filter the traffic.

Using the PAM table, you simply pick an existing application or define a new one for inspection thereby simplifying ACL configuration.

ICMP Inspection

An ICMP inspection session is on the basis of the source address of the inside host that originates the ICMP packet. Dynamic access control lists (ACLs) are created for return ICMP packets of the allowed types (echo-reply, time-exceeded, destination unreachable, and timestamp reply) for each session. There are no port numbers associated with an ICMP session, and the permitted IP address of the return packet is wild-carded in the ACL. The wildcard address is because the IP address of the return packet cannot be known in advance for time-exceeded and destination-unreachable replies. These replies can come from intermediate devices rather than the intended destination.

Application-Layer Protocol Inspection

In general, if you configure inspection for an application-layer protocol, packets for that protocol should be permitted to exit the firewall (by configuring the correct access control list), and packets for that protocol will only be allowed back in through the firewall if they belong to a valid existing session. Each protocol packet is inspected to maintain information about the session state.

Java, H.323, RPC, SIP, and SMTP inspection have additional information, described in the next five sections. [Table 3](#) lists the supported application-layer protocols.

Table 3 Protocol Keywords—Application-Layer Protocols

Protocol	Keyword
Application Firewall	appfw
CU-SeeMe	cuseeme
ESMTP	smtp
FTP	ftp
IMAP	imap
Java	http
H.323	h323
Microsoft NetShow	netshow
POP3	pop3
RealAudio	realaudio
RPC	rpc
SIP	sip
Simple Mail Transfer Protocol (SMTP)	smtp
Skinny Client Control Protocol (SCCP)	skinny
StreamWorks	streamworks
Structured Query Language*Net (SQL*Net)	sqlnet
TFTP	tftp
UNIX R commands (rlogin, rexec, rsh)	rcmd
VDOLive	vdolive
WORD	user-defined application name; use prefix -user
	Note All applications that appear under the show ip port-map command are supported.

Java Inspection

Java inspection enables Java applet filtering at the firewall. Java applet filtering distinguishes between trusted and untrusted applets by relying on a list of external sites that you designate as “friendly.” If an applet is from a friendly site, the firewall allows the applet through. If the applet is not from a friendly site, the applet will be blocked. Alternately, you could permit applets from all sites except sites specifically designated as “hostile.”

**Note**

Before you configure Java inspection, you must configure a numbered standard access list that defines “friendly” and “hostile” external sites. You configure this numbered standard access list to permit traffic from friendly sites, and to deny traffic from hostile sites. If you do not configure a numbered standard access list, but use a “placeholder” access list in the **ip inspect name** *inspection-name* **http** command, all Java applets will be blocked.

**Note**

Java blocking forces a strict order on TCP packets. To properly verify that Java applets are not in the response, a firewall will drop any TCP packet that is out of order. Because the network—not the firewall—determines how packets are routed, the firewall cannot control the order of the packets; the firewall can only drop and retransmit all TCP packets that are not in order.

**Caution**

Context-Based Access Control (CBAC) does not detect or block encapsulated Java applets. Therefore, Java applets that are wrapped or encapsulated, such as applets in .zip or .jar format, are *not* blocked at the firewall. CBAC also does not detect or block applets loaded via FTP, gopher, or HTTP on a nonstandard port.

H.323 Inspection

If you want CBAC inspection to work with NetMeeting 2.0 traffic (an H.323 application-layer protocol), you must also configure inspection for TCP, as described in the chapter “Configuring Context-Based Access Control” in the *Cisco IOS Security Configuration Guide*. This requirement exists because NetMeeting 2.0 uses an additional TCP channel not defined in the H.323 specification.

RPC Inspection

RPC inspection allows the specification of various program numbers. You can define multiple program numbers by creating multiple entries for RPC inspection, each with a different program number. If a program number is specified, all traffic for that program number will be permitted. If a program number is not specified, all traffic for that program number will be blocked. For example, if you created an RPC entry with the NFS program number, all NFS traffic will be allowed through the firewall.

SIP Inspection

You can configure SIP inspection to permit media sessions associated with SIP-signaled calls to traverse the firewall. Because SIP is frequently used to signal both incoming and outgoing calls, it is often necessary to configure SIP inspection in both directions on a firewall (both from the protected internal network and from the external network). Because inspection of traffic from the external network is not done with most protocols, it may be necessary to create an additional inspection rule to cause only SIP inspection to be performed on traffic coming from the external network.

SMTP Inspection

SMTP inspection causes SMTP commands to be inspected for illegal commands. Packets with illegal commands are modified to a “xxxx” pattern and forwarded to the server. This process causes the server to send a negative reply, forcing the client to issue a valid command. An illegal SMTP command is any command except the following:

- DATA
- HELO
- HELP

- MAIL
- NOOP
- QUIT
- RCPT
- RSET
- SAML
- SEND
- SOML
- VRFY

ESMTP Inspection

Like SMTP, ESMTP inspection also causes the commands to be inspected for illegal commands. Packets with illegal commands are modified to a “xxxx” pattern and forwarded to the server. This process causes the server to send a negative reply, forcing the client to issue a valid command. An illegal ESMTP command is any command except the following:

- AUTH
- DATA
- EHLO
- ETRN
- HELO
- HELP
- MAIL
- NOOP
- QUIT
- RCPT
- RSET
- SAML
- SEND
- SOML
- VRFY

In addition to inspecting commands, the ESMTP firewall also inspects the following extensions via deeper command inspection:

- Message Size Declaration (SIZE)
- Remote Queue Processing Declaration (ETRN)
- Binary MIME (BINARYMIME)
- Command Pipelining
- Authentication
- Delivery Status Notification (DSN)

- Enhanced Status Code (ENHANCEDSTATUSCODE)
- 8bit-MIMEtransport (8BITMIME)

**Note**

SMTP and ESMTP cannot exist simultaneously. An attempt to configure both protocols will result in an error message.

Use of the urlfilter Keyword

If you specify the **urlfilter** keyword, the Cisco IOS Firewall will interact with a URL filtering software to control web traffic for a given host or user on the basis of a specified security policy.

**Note**

Enabling HTTP inspection with or without any option triggers the Java applet scanner, which is CPU intensive. The only way to stop the Java applet scanner is to specify the **java-list access-list** option. Configuring URL filtering without enabling the **java-list access-list** option will severely impact performance.

Use of the timeout Keyword

If you specify a timeout for any of the transport-layer or application-layer protocols, the timeout will override the global idle timeout for the interface to which the set of inspection rules is applied.

If the protocol is TCP or a TCP application-layer protocol, the timeout will override the global TCP idle timeout. If the protocol is UDP or a UDP application-layer protocol, the timeout will override the global UDP idle timeout.

If you do not specify a timeout for a protocol, the timeout value applied to a new session of that protocol will be taken from the corresponding TCP or UDP global timeout value valid at the time of session creation.

The default ICMP timeout is deliberately short (10 seconds) due to the security hole that is opened by allowing ICMP packets with a wild-carded source address back into the inside network. The timeout will occur 10 seconds after the last outgoing packet from the originating host. For example, if you send a set of 10 ping packets spaced one second apart, the timeout will expire in 20 seconds or 10 seconds after the last outgoing packet. However, the timeout is not extended for return packets. If a return packet is not seen within the timeout window, the hole will be closed and the return packet will not be allowed in. Although the default timeout can be made longer if desired, it is recommended that this value be kept relatively short.

IP Fragmentation Inspection

CBAC inspection rules can help protect hosts against certain denial-of-service attacks involving fragmented IP packets. Even though the firewall keeps an attacker from making actual connections to a given host, the attacker may still be able to disrupt services provided by that host. This is done by sending many noninitial IP fragments or by sending complete fragmented packets through a router with an ACL that filters the first fragment of a fragmented packet. These fragments can tie up resources on the target host as it tries to reassemble the incomplete packets.

Using fragmentation inspection, the firewall maintains an *interfragment state* (structure) for IP traffic. Noninitial fragments are discarded unless the corresponding initial fragment was permitted to pass through the firewall. Noninitial fragments received before the corresponding initial fragments are discarded.

**Note**

Fragmentation inspection can have undesirable effects in certain cases, because it can result in the firewall discarding any packet whose fragments arrive out of order. There are many circumstances that can cause out-of-order delivery of legitimate fragments. Apply fragmentation inspection in situations where legitimate fragments, which are likely to arrive out of order, might have a severe performance impact.

Because routers running Cisco IOS software are used in a very large variety of networks, and because the CBAC feature is often used to isolate parts of internal networks from one another, the fragmentation inspection feature is not enabled by default. Fragmentation detection must be explicitly enabled for an inspection rule using the **ip inspect name** command. Unfragmented traffic is never discarded because it lacks a fragment state. Even when the system is under heavy attack with fragmented packets, legitimate fragmented traffic, if any, will still get some fraction of the firewall's fragment state resources, and legitimate, unfragmented traffic can flow through the firewall unimpeded.

Application Firewall Provisioning

Application firewall provisioning allows you to configure your Cisco IOS Firewall to detect and prohibit a specific protocol type of traffic.

Most firewalls provide only packet filtering capabilities that simply permit or deny traffic without inspecting the data stream; the Cisco IOS application firewall can detect whether or not a packet is in compliance with given HTTP protocol. If the packet is determined to be unauthorized, it will be dropped, the connection will be reset, and a syslog message will be generated, as appropriate.

User-Defined Applications

You can define your own applications and enter them into the port-to-application mapping (PAM) table using the **ip port-map** command. Then you set up your inspection rules by inserting your user-defined application as a value for the *protocol* argument in the **ip inspect name** command.

Session Limiting

Users can limit the number of established firewall sessions that a firewall rule creates by setting the "max-sessions" threshold. A session counter is maintained for each firewall interface. When a session count exceeds the specified threshold, an alert FW-4-SESSION_THRESHOLD_EXCEEDED message is logged to the syslog server and no new sessions can be created.

Examples

The following example causes the software to inspect TCP sessions and UDP sessions, and to specifically allow CU-SeeMe, FTP, and RPC traffic back through the firewall for existing sessions only. For UDP traffic, audit-trail is on. For FTP traffic, the idle timeout is set to override the global TCP idle timeout. For RPC traffic, program numbers 100003, 100005, and 100021 are permitted.

```
ip inspect name myrules tcp
ip inspect name myrules udp audit-trail on
ip inspect name myrules cuseeme
ip inspect name myrules ftp timeout 120
ip inspect name myrules rpc program-number 100003
ip inspect name myrules rpc program-number 100005
ip inspect name myrules rpc program-number 100021
```

The following example adds fragment checking to software inspection of TCP and UDP sessions for the rule named “myrules.” In this example, the firewall software will allocate 100 state structures, and the timeout value for dropping unassembled packets is set to 4 seconds. If 100 initial fragments for 100 different packets are sent through the router, all of the state structures will be used up. The initial fragment for packet 101 will be dropped. Additionally, if the number of free state structures (structures available for use by unassembled packets) drops below the threshold values, 32 or 16, the timeout value is automatically reduced to 2 or 1, respectively. Changing the timeout value frees up packet state structures more quickly.

```
ip inspect name myrules tcp
ip inspect name myrules udp audit-trail on
ip inspect name myrules cuseeme
ip inspect name myrules ftp timeout 120
ip inspect name myrules rpc program-number 100003
ip inspect name myrules rpc program-number 100005
ip inspect name myrules rpc program-number 100021
ip inspect name myrules fragment max 100 timeout 4
```

The following firewall and SIP example shows how to allow outside-initiated calls and internal calls. For outside-initiated calls, an ACL needs to be punched to allow for the traffic from the initial signaling packet from outside. Subsequent signaling and media channels will be allowed by the inspection module.

```
ip inspect name voip sip
interface FastEthernet0/0
 ip inspect voip in
!
!
interface FastEthernet0/1
 ip inspect voip in
 ip access-group 100 in
!
!
access-list 100 permit udp host <gw ip> any eq 5060
access-list 100 permit udp host <proxy ip> any eq 5060
access-list deny ip any any
```

The following example shows two configured inspections named fw_only and fw_urlf; URL filtering will work only on the traffic that is inspected by fw_urlf. Note that the **java-list access-list** option has been enabled, which disables java scanning.

```
ip inspect name fw_only http java-list 51 timeout 30
interface e0
 ip inspect fw_only in
!
ip inspect name fw_urlf http urlfilter java-list 51 timeout 30
interface e1
 ip inspect fw_urlf in
```

The following example shows how to define the HTTP application firewall policy mypolicy. This policy includes all supported HTTP policy rules. This example also includes sample output from the **show appfw configuration** and **show ip inspect config** commands, which allow you to verify the configured setting for the application policy.

```
! Define the HTTP policy.
appfw policy-name mypolicy
 application http
  strict-http action allow alarm
  content-length maximum 1 action allow alarm
  content-type-verification match-req-rsp action allow alarm
  max-header-length request 1 response 1 action allow alarm
  max-uri-length 1 action allow alarm
  port-misuse default action allow alarm
```

```

request-method rfc default action allow alarm
request-method extension default action allow alarm
transfer-encoding type default action allow alarm
!
!
! Apply the policy to an inspection rule.
ip inspect name firewall appfw mypolicy
ip inspect name firewall http
!
!
! Apply the inspection rule to all HTTP traffic entering the FastEthernet0/0 interface.
interface FastEthernet0/0
 ip inspect firewall in
!
!
! Issue the show appfw configuration command and the show ip inspect config command after
the inspection rule "mypolicy" is applied to all incoming HTTP traffic on the
FastEthernet0/0 interface.
!
Router# show appfw configuration

Application Firewall Rule configuration
  Application Policy name mypolicy
    Application http
      strict-http action allow alarm
      content-length minimum 0 maximum 1 action allow alarm
      content-type-verification match-req-rsp action allow alarm
      max-header-length request length 1 response length 1 action allow alarm
      max-uri-length 1 action allow alarm
      port-misuse default action allow alarm
      request-method rfc default action allow alarm
      request-method extension default action allow alarm
      transfer-encoding default action allow alarm

Router# show ip inspect config

Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
Inspection name firewall
http alert is on audit-trail is off timeout 3600

```

Related Commands

Command	Description
ip inspect	Applies a set of inspection rules to an interface.
ip inspect alert-off	Disables CBAC alert messages.
ip inspect audit trail	Turns on CBAC audit trail messages, which will be displayed on the console after each CBAC session close.

show ip inspect

To display Context-Based Access Control (CBAC) configuration and session information, use the **show ip inspect** command in privileged EXEC mode.

```
show ip inspect { name inspection-name | config | interfaces | session [detail] | statistics | all } [vrf
vrf-name]
```

Syntax Description

name <i>inspection-name</i>	Displays the configured inspection rule with the name <i>inspection-name</i> .
config	Displays the complete CBAC inspection configuration.
interfaces	Displays the interface configuration with respect to applied inspection rules and access lists.
session [detail]	Displays existing sessions that are currently being tracked and inspected by CBAC. The optional detail keyword allows additional details about these sessions to be shown.
statistics	Displays CBAC sessions statistics, such as the number of TCP and HTTP packets that are processed through the inspection, the number of sessions that have been created since the subsystem startup, the current session count, the maximum session count, and the session creation rate.
all	Displays all CBAC configuration and all existing sessions that are currently being tracked and inspected by CBAC.
vrf <i>vrf-name</i>	(Optional) Displays information only for the specified Virtual Routing and Forwarding (VRF) interface.

Command Modes

Privileged EXEC

Command History

Release	Modification
11.2 P	This command was introduced.
12.3(4)T	The output for the show ip inspect session detail command was enhanced to support dynamic access control list (ACL) bypass.
12.3(11)T	The statistics keyword was added.
12.3(14)T	The output shows the IMAP and POP3 configuration. The vrf vrf-name keyword/argument pair was added.

Usage Guidelines

Use this command to view the CBAC configuration and session information.

ACL Bypass Functionality

ACL bypass allows a packet to avoid redundant ACL checks by allowing the firewall to permit the packet on the basis of existing inspection sessions instead of dynamic ACLs. Because input and output dynamic ACLs have been eliminated from the firewall configuration, the **show ip inspect session detail** command output no longer shows dynamic ACLs. Instead, the output displays the matching inspection session for each packet that is permitted through the firewall.

Examples

The following example shows sample output for the **show ip inspect name myinspectionrule** command, where the inspection rule “myinspectionrule” is configured. In this example, the output shows the protocols that should be inspected by CBAC and the corresponding idle timeouts for each protocol.

```
Router# show ip inspect name myinspectionrule
```

```
Inspection Rule Configuration
  Inspection name myinspectionrule
    tcp timeout 3600
    udp timeout 30
    ftp timeout 3600
```

The following is sample output for the **show ip inspect config** command. In this example, the output shows CBAC configuration, including global timeouts, thresholds, and inspection rules.

```
Session audit trail is disabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
  Inspection name myinspectionrule
    tcp timeout 3600
    udp timeout 30
    ftp timeout 3600
```

The following is sample output for the **show ip inspect interfaces** command:

```
Interface Configuration
Interface Ethernet0
  Inbound inspection rule is myinspectionrule
    tcp timeout 3600
    udp timeout 30
    ftp timeout 3600
  Outgoing inspection rule is not set
  Inbound access list is not set
  Outgoing access list is not set
```

The following is sample output for the **show ip inspect session** command. In this example, the output shows the source and destination addresses and port numbers (separated by colons), and it indicates that the session is an FTP session.

```
Router# show ip inspect session
```

```
Established Sessions
  Session 25A3318 (10.0.0.1:20)=>(10.1.0.1:46068) ftp-data SIS_OPEN
  Session 25A6E1C (10.1.0.1:46065)=>(10.0.0.1:21) ftp SIS_OPEN
```

The following is sample output for the **show ip inspect all** command:

```
Router# show ip inspect all
```

```
Session audit trail is disabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
  Inspection name all
```

```

    tcp timeout 3600
    udp timeout 30
    ftp timeout 3600
Interface Configuration
Interface Ethernet0
  Inbound inspection rule is all
    tcp timeout 3600
    udp timeout 30
    ftp timeout 3600
  Outgoing inspection rule is not set
  Inbound access list is not set
  Outgoing access list is not set
Established Sessions
Session 25A6E1C (30.0.0.1:46065)=>(40.0.0.1:21) ftp SIS_OPEN
Session 25A34A0 (40.0.0.1:20)=>(30.0.0.1:46072) ftp-data SIS_OPEN

```

The following is sample output from the **show ip inspect session detail** command, which shows that an outgoing ACL and an inbound ACL (dynamic ACLs) have been created to allow return traffic:

```

Router# show ip inspect session detail

Established Sessions
Session 80E87274 (192.168.1.116:32956)=>(192.168.101.115:23) tcp SIS_OPEN
  Created 00:00:08, Last heard 00:00:04
  Bytes sent (initiator:responder) [140:298] acl created 2
  Outgoing access-list 102 applied to interface FastEthernet0/0
  Inbound access-list 101 applied to interface FastEthernet0/1

```

The following is sample output from the **show ip inspect session detail** command, which shows related ACL information (such as session identifiers [SIDs]), but does not show dynamic ACLs, which are no longer created:

```

Router# show ip inspect session detail

Established Sessions
Session 814063CC (192.168.1.116:32955)=>(192.168.101.115:23) tcp SIS_OPEN
  Created 00:00:10, Last heard 00:00:06
  Bytes sent (initiator:responder) [140:298]
  In SID 192.168.101.115[23:23]=>192.168.1.117[32955:32955] on ACL 101 (15 matches)
  Out SID 192.168.101.115[23:23]=>192.168.1.116[32955:32955] on ACL 102

```

The following is sample output from the **show ip inspect statistics** command:

```

Router# show ip inspect statistics

Packet inspection statistics [process switch:fast switch]
  tcp packets: [616668:0]
  http packets: [178912:0]
Interfaces configured for inspection 1
Session creations since subsystem startup or last reset 42940
Current session counts (estab/half-open/terminating) [0:0:0]
Maxever session counts (estab/half-open/terminating) [98:68:50]
Last session created 5d21h
Last statistic reset never
Last session creation rate 0
Last half-open session total 0

Router#

```

Glossary

authentication—Process during which any unrecognized command causes the Cisco IOS Firewall to drop the packet and close the connection.

CBAC—Context-Based Access Control. A Cisco IOS Firewall set feature that scrutinizes source and destination addresses to enhance security for TCP and UDP applications that use well-known ports, such as FTP and e-mail traffic.

ESMTP—Extended Simple Mail Transfer Protocol. An extended version of the Simple Mail Transfer Protocol (SMTP), which includes additional functionality, such as delivery notification and session delivery.

IMAP—Internet Message Access Protocol. A method of accessing e-mail or bulletin board messages kept on a mail server that can be shared. IMAP permits client e-mail applications to access remote message stores as if they were local without actually transferring the message.

POP—Post Office Protocol. A protocol that client e-mail applications use to retrieve mail from a mail server.

SMTP—Simple Mail Transfer Protocol. An Internet protocol providing e-mail services.

SSL—Secure Socket Layer Protocol. This protocol is used to deliver secure information over the Internet.

state router—A router that tracks the client/server commands until the client is authenticated.

TCP—Transmission Control Protocol. A connection-oriented transport layer protocol that provides reliable full-duplex data transmission. TCP is part of the TCP/IP protocol stack.

UDP—User Datagram Protocol. A connectionless transport-layer protocol for exchanging datagrams without acknowledgments or guaranteed delivery.

VPN—Virtual Private Network. A network that enables IP traffic to travel securely over a public TCP/IP network by encrypting all traffic from one network to another. A VPN network uses “tunneling” to encrypt all information at the IP level.



Note

Refer to [Networking Terms and Acronyms](#) for terms not included in this glossary.

CCVP, the Cisco logo, and Welcome to the Human Network are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn is a service mark of Cisco Systems, Inc.; and Access Registrar, Aironet, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unity, Enterprise/Solver, EtherChannel, EtherFast, EtherSwitch, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, IOS, iPhone, IP/TV, iQ Expertise, the iQ logo, iQ Net Readiness Scorecard, iQuick Study, LightStream, Linksys, MeetingPlace, MGX, Networkers, Networking Academy, Network Registrar, PIX, ProConnect, ScriptShare, SMARTnet, StackWise, The Fastest Way to Increase Your Internet Quotient, and TransPath are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or Website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (0711R)

