Security Configuration Guide: Context-Based Access Control Firewall, Cisco IOS Release 15M&T

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: http://www.cisco.com/go/trademarks. Third-party trademarks mentioned 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. (1110R)

CONTENTS

CHAPTER 1

Configuring Context-Based Access Control

This chapter describes how to configure Context-based Access Control (CBAC). CBAC provides advanced traffic filtering functionality and can be used as an integral part of your network's firewall. For more information regarding firewalls, refer to the chapter "Cisco IOS Firewall Overview."

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Configuring Context-Based Access Control

- If you try to configure Context-based Access Control (CBAC) but do not have a good understanding of how CBAC works, you might inadvertently introduce security risks to the firewall and to the protected network. You should be sure you understand what CBAC does before you configure CBAC.
- As with all networking devices, protect access into the firewall by configuring passwords as described in the "Configuring Passwords and Privileges" chapter. You should also consider configuring user authentication, authorization, and accounting as described in the "Authentication, Authorization, and

Accounting (AAA)” part of this guide. Additional guidelines to help you establish a good security policy can be found in the “Cisco IOS Firewall Overview” chapter.

Restrictions for Configuring Context-Based Access Control

The following restrictions apply to Context-Based Access Control (CBAC) firewalls:

- Supports only TCP and UDP IP protocol traffic. Other IP traffic, such as Internet Control Message Protocol (ICMP), is not inspected by CBAC and should be filtered with basic access lists.
- Does not inspect IPv6 router-generated traffic.
- Does not inspect CNA packets that have firewall as the source or destination address.
- Ignores ICMP Unreachable messages.
- H.323 Version 2 and Real Time Streaming Protocol (RTSP) inspection supports only the following multimedia client-server applications such as Cisco IPTV, RealNetworks RealAudio G2 Player, Apple QuickTime 4.
- While configuring CBAC, if you reconfigure access lists to block TFTP traffic into an interface, you will not be able to netboot over that interface. This is not a CBAC-specific limitation, but is part of existing access list functionality.
- When you configure inspect rules on both ingress and egress interfaces and the protocol configured in the egress inspect rule is not present in the ingress inspect rule, the traffic in the egress direction is not inspected. However, traffic is inspected on both ingress and egress interfaces if the protocol is configured on both interfaces and if the protocol is configured only on the egress interface.

FTP Traffic and CBAC

- With FTP, CBAC does not allow third-party connections (three-way FTP transfer).
- When CBAC inspects FTP traffic, it only allows data channels with the destination port in the range of 1024 to 65535.
- CBAC will not open a data channel if the FTP client-server authentication fails.

IPSec and CBAC Compatibility

When CBAC and IPSec are enabled on the same router, and the firewall router is an endpoint for IPSec for the particular flow, then IPSec is compatible with CBAC (that is, CBAC can do its normal inspection processing on the flow).

If the router is not an IPSec endpoint, but the packet is an IPSec packet, then CBAC will not inspect the packets because the protocol number in the IP header of the IPSec packet is not TCP or UDP. CBAC only inspects UDP and TCP packets.

Information About Context-Based Access Control

What CBAC Does

CBAC works to provide network protection on multiple levels using the following functions:

Traffic Filtering

CBAC intelligently filters TCP and UDP packets based on application-layer protocol session information. You can configure CBAC to permit specified TCP and UDP traffic through a firewall only when the connection is initiated from within the network you want to protect. CBAC can inspect traffic for sessions that originate from either side of the firewall, and CBAC can be used for intranet, extranet, and Internet perimeters of your network.

Without CBAC, traffic filtering is limited to access list implementations that examine packets at the network layer, or at most, the transport layer. However, CBAC examines not only network layer and transport layer information but also examines the application-layer protocol information (such as FTP connection information) to learn about the state of the session. This allows support of protocols that involve multiple channels created as a result of negotiations in the control channel. Most of the multimedia protocols as well as some other protocols (such as FTP, RPC, and SQL*Net) involve multiple channels.

Using CBAC, Java blocking can be configured to filter HTTP traffic based on the server address or to completely deny access to Java applets that are not embedded in an archived or compressed file. With Java, you must protect against the risk of users inadvertently downloading destructive applets into your network. To protect against this risk, you could require all users to disable Java in their browser. If this is not an acceptable solution, you can create a CBAC inspection rule to filter Java applets at the firewall, which allows users to download only applets residing within the firewall and trusted applets from outside the firewall. For extensive content filtering of Java, Active-X, or virus scanning, you might want to consider purchasing a dedicated content filtering product.

Traffic Inspection

CBAC inspects traffic that travels through the firewall to discover and manage state information for TCP and 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.

CBAC helps to protect against DoS attacks in other ways. CBAC inspects packet sequence numbers in TCP connections to see if they are within expected ranges--CBAC drops any suspicious packets. You can also configure CBAC to drop half-open connections, which require firewall processing and memory resources to maintain. Additionally, CBAC can detect unusually high rates of new connections and issue alert messages.

CBAC can help by protecting against certain DoS attacks involving fragmented IP packets. Even though the firewall prevents an attacker from making actual connections to a given host, the attacker can disrupt services provided by that host. This is done by sending many non-initial 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.

Alerts and Audit Trails

CBAC also generates real-time alerts and audit trails. Enhanced audit trail features use SYSLOG to track all network transactions; recording time stamps, source host, destination host, ports used, and the total number of transmitted bytes, for advanced, session-based reporting. Real-time alerts send SYSLOG error messages to central management consoles upon detecting suspicious activity. Using CBAC inspection rules, you can configure alerts and audit trail information on a per-application protocol basis. For example, if you want to generate audit trail information for HTTP traffic, you can specify that in the CBAC rule covering HTTP inspection.

Intrusion Prevention

CBAC provides a limited amount of intrusion detection to protect against specific SMTP attacks. With intrusion detection, SYSLOG messages are reviewed and monitored for specific “attack signatures.” Certain types of network attacks have specific characteristics, or signatures. When CBAC detects an attacks, it resets the offending connections and sends SYSLOG information to the SYSLOG server. Refer to the section "CBAC Configuration Examples" later in this chapter for a list of supported signatures.

In addition to the limited intrusion detection offered by CBAC, the Cisco IOS Firewall feature set offers intrusion detection technology for mid-range and high-end router platforms using the Cisco IOS Intrusion Prevention System (IPS). Cisco IOS IPS restructures and replaces the existing Cisco IOS Intrusion Detection System (IDS). It is ideal for any network perimeter, and especially for locations in which a router is being deployed and additional security between network segments is required. It also can protect intranet and extranet connections where additional security is mandated, and branch-office sites connecting to the corporate office or Internet.

The Cisco IOS IPS acts as an in-line intrusion detection sensor, watching packets and sessions as they flow through the router and scanning each packet to match any of the Cisco IOS IPS signatures. When Cisco IOS IPS detects suspicious activity, it responds before network security can be compromised and logs the event through Cisco IOS syslog messages or Security Device Event Exchange (SDEE).

For more information about Cisco IOS IPS, refer to the module “Configuring Cisco IOS Intrusion Prevention System (IPS).”

What CBAC Does Not Do

CBAC does not provide intelligent filtering for all protocols; it only works for the protocols that you specify. If you do not specify a certain protocol for CBAC, the existing access lists will determine how that protocol is filtered. No temporary openings will be created for protocols not specified for CBAC inspection.

CBAC does not protect against attacks originating from within the protected network unless that traffic travels through a router that has the Cisco IOS Firewall feature set deployed on it. CBAC only detects and protects against attacks that travel through the firewall. This is a scenario in which you might want to deploy CBAC on an intranet-based router.

CBAC protects against certain types of attacks, but not every type of attack. CBAC should not be considered a perfect, impenetrable defense. Determined, skilled attackers might be able to launch effective attacks. While there is no such thing as a perfect defense, CBAC detects and prevents most of the popular attacks on your network.

How CBAC Works-Overview

CBAC creates temporary openings in access lists at firewall interfaces. These openings are created when specified traffic exits your internal network through the firewall. The openings allow returning traffic (that would normally be blocked) and additional data channels to enter your internal network back through the firewall. The traffic is allowed back through the firewall only if it is part of the same session as the original traffic that triggered CBAC when exiting through the firewall.

Throughout this chapter, the terms "inbound" and "outbound" are used to describe the direction of traffic relative to the router interface on which CBAC is applied. For example, if a CBAC rule is applied inbound on interface E0, then packets entering interface E0 from the network will be inspected. If a CBAC rule is applied outbound on interface E0, then packets leaving interface E0 to the network will be inspected. This is similar to the way ACLs work.

For example, consider a CBAC inspection rule named hqusers, and suppose that rule is applied inbound at interface E0:

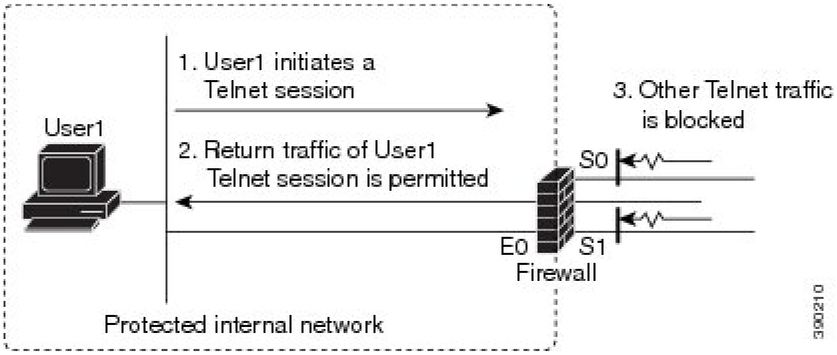
```
Device(config-if)# ip inspect hqusers in
```

This command causes CBAC to inspect the packets coming into this interface from the network. If a packet is attempting to initiate a session, CBAC will then determine if this protocol is allowed, create a CBAC session, add the appropriate ACLs to allow return traffic and do any needed content inspection on any future packets for this session.

The terms "input" and "output" are used to describe the interfaces at which network traffic enters or exits the firewall router. A packet enters the firewall router via the input interface, is inspected by the firewall software and then exits the router via the output interface.

In the figure below, the inbound access lists at S0 and S1 are configured to block Telnet traffic, and there is no outbound access list configured at E0. When the connection request for User1's Telnet session passes through the firewall, CBAC creates a temporary opening in the inbound access list at S0 to permit returning Telnet traffic for User1's Telnet session. (If the same access list is applied to both S0 and S1, the same opening would appear at both interfaces.) If necessary, CBAC would also have created a similar opening in an outbound access list at E0 to permit return traffic.

Figure 1: CBAC Opens Temporary Holes in Firewall Access Lists



How CBAC Works-Details

This section describes how CBAC inspects packets and maintains state information about sessions to provide intelligent filtering.

Packets Are Inspected

With CBAC, you specify which protocols you want to be inspected, and you specify an interface and interface direction (in or out) where inspection originates. Only specified protocols will be inspected by CBAC.

Packets entering the firewall are inspected by CBAC only if they first pass the inbound access list at the input interface and outbound access list at the output interface. If a packet is denied by the access list, the packet is simply dropped and not inspected by CBAC.

CBAC inspection tracks sequence numbers in all TCP packets, and drops those packets with sequence numbers that are not within expected ranges.

CBAC inspection recognizes application-specific commands (such as illegal SMTP commands) in the control channel, and detects and prevents certain application-level attacks.

When CBAC suspects an attack, the DoS feature can take several actions:

- Generate alert messages
- Protect system resources that could impede performance
- Block packets from suspected attackers

CBAC uses timeout and threshold values to manage session state information, helping to determine when to drop sessions that do not become fully established. Setting timeout values for network sessions helps prevent DoS attacks by freeing up system resources, dropping sessions after a specified amount of time. Setting threshold values for network sessions helps prevent DoS attacks by controlling the number of half-open sessions, which limits the amount of system resources applied to half-open sessions. When a session is dropped, CBAC sends a reset message to the devices at both end points (source and destination) of the session. When the system under DoS attack receives a reset command, it releases, or frees up, processes and resources related to that incomplete session.

CBAC provides three thresholds against DoS attacks:

- The total number of half-open TCP or UDP sessions
- The number of half-open sessions based upon time
- The number of half-open TCP-only sessions per host

If a threshold is exceeded, CBAC has two options:

- Send a reset message to the end points of the oldest half-open session, making resources available to service newly arriving SYN packets.
- In the case of half open TCP only sessions, CBAC blocks all SYN packets temporarily for the duration configured by the threshold value. When the router blocks a SYN packet, the TCP three-way handshake is never initiated, which prevents the router from using memory and processing resources needed for valid connections.

DoS detection and prevention requires that you create a CBAC inspection rule and apply that rule on an interface. The inspection rule must include the protocols that you want to monitor against DoS attacks. For example, if you have TCP inspection enabled on the inspection rule, then CBAC can track all TCP connections to watch for DoS attacks. If the inspection rule includes FTP protocol inspection but not TCP inspection, CBAC tracks only FTP connections for DoS attacks.

For detailed information about setting timeout and threshold values in CBAC to detect and prevent DoS attacks, refer in the "How to Configure Context-Based Access Control" section.

A State Table Maintains Session State Information

Whenever a packet is inspected, a state table is updated to include information about the state of the session.

Return traffic will only be permitted back through the firewall if the state table contains information indicating that the packet belongs to a permissible session. CBAC controls the traffic that belongs to a valid session. When return traffic is inspected, the state table information is updated as necessary.

UDP Sessions Are Approximated

With UDP--a connectionless service--there are no actual sessions, so the software approximates sessions by examining the information in the packet and determining if the packet is similar to other UDP packets (for example, same source/destination addresses and port numbers) and if the packet was detected soon after another similar UDP packet. “Soon” means within the configurable UDP idle timeout period.

Access List Entries Are Dynamically Created and Deleted to Permit Return Traffic and Additional Data Connections

CBAC dynamically creates and deletes access list entries at the firewall interfaces, according to the information maintained in the state tables. These access list entries are applied to the interfaces to examine traffic flowing back into the internal network. These entries create temporary openings in the firewall to permit only traffic that is part of a permissible session.

The temporary access list entries are never saved to NVRAM.

When and Where to Configure CBAC

Configure CBAC at firewalls protecting internal networks. Such firewalls should be Cisco routers with the Cisco IOS Firewall feature set configured as described previously in the section “Cisco IOS Firewall.”

Use CBAC when the firewall will be passing traffic such as the following:

- Standard TCP and UDP Internet applications
- Multimedia applications
- Oracle support

Use CBAC for these applications if you want the application’s traffic to be permitted through the firewall only when the traffic session is initiated from a particular side of the firewall (usually from the protected internal network).

In many cases, you will configure CBAC in one direction only at a single interface, which causes traffic to be permitted back into the internal network only if the traffic is part of a permissible (valid, existing) session. This is a typical configuration for protecting your internal networks from traffic that originates on the Internet.

You can also configure CBAC in two directions at one or more interfaces. CBAC is configured in two directions when the networks on both sides of the firewall should be protected, such as with extranet or intranet configurations, and to protect against DoS attacks. For example, if the firewall is situated between two partner companies' networks, you might wish to restrict traffic in one direction for certain applications, and restrict traffic in the opposite direction for other applications.

The CBAC Process

This section describes a sample sequence of events that occurs when CBAC is configured at an external interface that connects to an external network such as the Internet.

In this example, a TCP packet exits the internal network through the firewall's external interface. The TCP packet is the first packet of a Telnet session, and TCP is configured for CBAC inspection.

1. The packet reaches the firewall's external interface.
2. The packet is evaluated against the interface's existing outbound access list, and the packet is permitted. (A denied packet would simply be dropped at this point.)
3. The packet is inspected by CBAC to determine and record information about the state of the packet's connection. This information is recorded in a new state table entry created for the new connection.

(If the packet's application--Telnet--was not configured for CBAC inspection, the packet would simply be forwarded out the interface at this point without being inspected by CBAC. See the section "Defining an Inspection Rule" later in this chapter for information about configuring CBAC inspection.)

1. Based on the obtained state information, CBAC creates a temporary access list entry which is inserted at the beginning of the external interface's inbound extended access list. This temporary access list entry is designed to permit inbound packets that are part of the same connection as the outbound packet just inspected.
2. The outbound packet is forwarded out the interface.
3. Later, an inbound packet reaches the interface. This packet is part of the same Telnet connection previously established with the outbound packet. The inbound packet is evaluated against the inbound access list, and it is permitted because of the temporary access list entry previously created.
4. The permitted inbound packet is inspected by CBAC, and the connection's state table entry is updated as necessary. Based on the updated state information, the inbound extended access list temporary entries might be modified in order to permit only packets that are valid for the current state of the connection.
5. Any additional inbound or outbound packets that belong to the connection are inspected to update the state table entry and to modify the temporary inbound access list entries as required, and they are forwarded through the interface.
6. When the connection terminates or times out, the connection's state table entry is deleted, and the connection's temporary inbound access list entries are deleted.

In the sample process just described, the firewall access lists are configured as follows:

- An outbound IP access list (standard or extended) is applied to the external interface. This access list permits all packets that you want to allow to exit the network, including packets you want to be inspected by CBAC. In this case, Telnet packets are permitted.
- An inbound extended IP access list is applied to the external interface. This access list denies any traffic to be inspected by CBAC--including Telnet packets. When CBAC is triggered with an outbound packet, CBAC creates a temporary opening in the inbound access list to permit only traffic that is part of a valid, existing session.

If the inbound access list had been configured to permit all traffic, CBAC would be creating pointless openings in the firewall for packets that would be permitted anyway.

CBAC Supported Protocols

You can configure CBAC to inspect the following types of sessions:

- All TCP sessions, regardless of the application-layer protocol (sometimes called “single-channel” or “generic” TCP inspection)
- All UDP sessions, regardless of the application-layer protocol (sometimes called “single-channel” or “generic” UDP inspection)

You can also configure CBAC to specifically inspect certain application-layer protocols. The following application-layer protocols can all be configured for CBAC:

- CU-SeeMe (only the White Pine version)
- FTP
- H.323 (such as NetMeeting, ProShare)
- HTTP (Java blocking)
- Microsoft NetShow
- UNIX R-commands (such as rlogin, rexec, and rsh)
- RealAudio
- RTSP (Real Time Streaming Protocol)
- RPC (Sun RPC, not DCE RPC)
- SMTP (Simple Mail Transport Protocol)

Note CBAC can be configured to inspect SMTP but not ESMTP (Extended Simple Mail Transport Protocol). SMTP is described in RFC 821. CBAC SMTP inspect does not inspect the ESMTP session or command sequence. Configuring SMTP inspection is not useful for ESMTP, and it can cause problems. To determine whether a mail-server is doing SMTP or ESMTP, contact your mail-server software vendor, or telnet to the mail-server port 25 and observe the banner to see if it reports SMTP or ESMTP.

- SQL*Net
- StreamWorks

- TFTP
- VDOLive

When a protocol is configured for CBAC, that protocol traffic is inspected, state information is maintained, and in general, packets are allowed back through the firewall only if they belong to a permissible session.

RTSP and H.323 Protocol Support for Multimedia Applications

CBAC supports a number of protocols for multimedia applications that require delivery of data with real-time properties such as audio and video conferencing. This support includes the following multimedia application protocols:

- Real Time Streaming Protocol (RTSP)
- H.323 Version 2 (H.323 V2)

RTSP and H.323 V2 inspection allows clients on a protected network to receive data associated with a multimedia session from a server on an unprotected network.

RTSP Support

RTSP is the IETF standards-based protocol (RFC 2326) for control over the delivery of data with real-time properties such as audio and video streams. It is useful for large-scale broadcasts and audio or video on demand streaming, and is supported by a variety of vendor products of streaming audio and video multimedia, including Cisco IP/TV, RealNetworks RealAudio G2 Player, and Apple QuickTime 4 software.

RFC 2326 allows RTSP to run over either UDP or TCP, though CBAC currently supports only TCP-based RTSP. RTSP establishes a TCP-based control connection, or channel, between the multimedia client and server. RTSP uses this channel to control commands such as “play” and “pause” between the client and server. These control commands and responses are text-based and are similar to HTTP.

RTSP typically relies on a UDP-based data transport protocol such as standard Real-Time Transport Protocol (RTP) to open separate channels for data and for RTP Control Protocol (RTCP) messages. RTP and RTCP channels occur in pairs, with RTP being an even numbered port and RTCP being the next consecutive port. Understanding the relationship of RTP and RTCP is important for verifying session information using CBAC **show** commands.

The RTSP client uses TCP port 554 or 8554 to open a multimedia connection with a server. The data channel or data control channel (using RTCP) between the client and the server is dynamically negotiated between the client and the server using any of the high UDP ports (1024 to 65536).

CBAC uses this port information along with connection information from the client to create dynamic access control list (ACL) entries in the firewall. As TCP or UDP connections are terminated, CBAC removes these dynamic entries from the appropriate ACLs.

CBAC support for RTSP includes the following data transport modes:

- Standard Real-Time Transport Protocol (RTP)

RTP is an IETF standard (RFC 1889) supporting delivery of real-time data such as audio and video. RTP uses the RTP Control Protocol (RTCP) for managing the delivery of the multimedia data stream. This is the normal mode of operation for Cisco IP/TV and Apple QuickTime 4 software.

- RealNetworks Real Data Transport (RDT)

RDT is a proprietary protocol developed by RealNetworks for data transport. This mode uses RTSP for communication control and uses RDT for the data connection and retransmission of lost packets. This is the normal mode of operation for the RealServer G2 from RealNetworks.

- Interleaved (Tunnel Mode)

In this mode, RTSP uses the control channel to tunnel RTP or RDT traffic.

- Synchronized Multimedia Integration Language (SMIL)

SMIL is a layout language that enables the creation of multimedia presentations consisting of multiple elements of music, voice, images, text, video and graphics. This involves multiple RTSP control and data streams between the player and the servers. This mode is available only using RTSP and RDT. SMIL is a proposed specification of the World Wide Web Consortium (W3C). The RealNetworks RealServer and RealServer G2 provide support for SMIL--Cisco IP/TV and Apple QuickTime 4 do not.

H.323 Support

CBAC support for H.323 inspection includes H.323 Version 2 and H.323 Version 1. H.323 V2 provides additional options over H.323 V1, including a “fast start” option. The fast start option minimizes the delay between the time that a user initiates a connection and the time that the user gets the data (voice, video). H.323 V2 inspection is backward compatible with H.323 V1.

With H.323 V1, after a TCP connection is established between the client and server (H.225 Channel), a separate channel for media control (H.245 Channel) is opened through which multimedia channels for audit and video are further negotiated.

The H.323 V2 client opens a connection to server which is listening on port 1720. The data channel between the client and the server is dynamically negotiated using any of the high UDP ports (1024 to 65536).

CBAC uses this port information along with connection information from the client to create dynamic access control list (ACL) entries in the firewall. As TCP or UDP connections are terminated, CBAC removes these dynamic entries from the appropriate ACLs.

Memory and Performance Impact

CBAC uses less than approximately 600 bytes of memory per connection. Because of the memory usage, you should use CBAC only when you need to. There is also a slight amount of additional processing that occurs whenever packets are inspected.

Sometimes CBAC must evaluate long access lists, which might have presented a negative impact to performance. However, this impact is avoided, because CBAC evaluates access lists using an accelerated method (CBAC hashes access lists and evaluates the hash).

Picking an Interface Internal or External

You must decide whether to configure CBAC on an internal or external interface of your firewall.

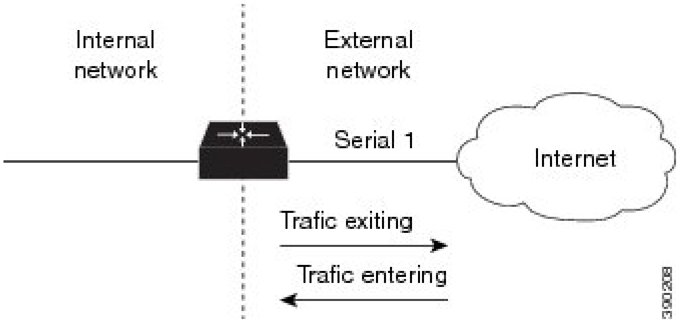
“Internal” refers to the side where sessions must originate for their traffic to be permitted through the firewall. “External” refers to the side where sessions cannot originate (sessions originating from the external side will be blocked).

If you will be configuring CBAC in two directions, you should configure CBAC in one direction first, using the appropriate “internal” and “external” interface designations. When you configure CBAC in the other direction, the interface designations will be swapped. (CBAC can be configured in two directions at one or more interfaces. Configure CBAC in two directions when the networks on both sides of the firewall require protection, such as with extranet or intranet configurations, and for protection against DoS attacks.)

The firewall is most commonly used with one of two basic network topologies. Determining which of these topologies is most like your own can help you decide whether to configure CBAC on an internal interface or on an external interface.

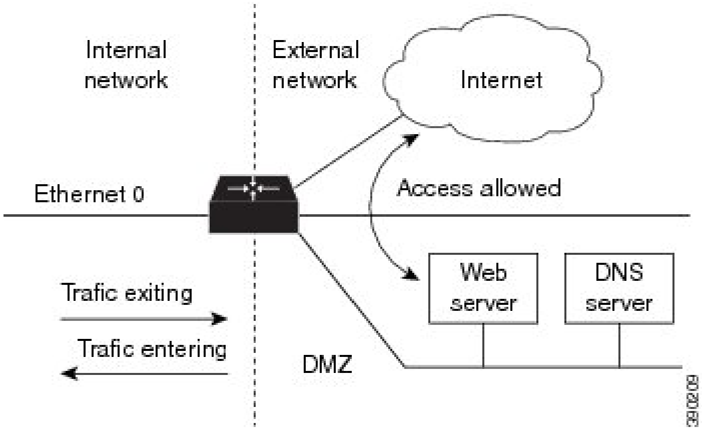
The first topology is shown in the figure below. In this simple topology, CBAC is configured for the external interface Serial 1. This prevents specified protocol traffic from entering the firewall and the internal network, unless the traffic is part of a session initiated from within the internal network.

Figure 2: Simple Topology--CBAC Configured at the External Interface



The second topology is shown in the figure below. In this topology, CBAC is configured for the internal interface Ethernet 0. This allows external traffic to access the services in the Demilitarized Zone (DMZ), such as DNS services, but prevents specified protocol traffic from entering your internal network--unless the traffic is part of a session initiated from within the internal network.

Figure 3: DMZ Topology--CBAC Configured at the Internal Interface



Using these two sample topologies, decide whether to configure CBAC on an internal or external interface.

To view various firewall configuration scenarios, see the "CBAC Configuration Examples" section at the end of this chapter.

Configuring IP Access Lists at the Interface

For CBAC to work properly, you need to make sure that you have IP access lists configured appropriately at the interface.

Follow these three general rules when evaluating your IP access lists at the firewall:

- Start with a basic configuration.

If you try to configure access lists without a good understanding of how access lists work, you might inadvertently introduce security risks to the firewall and to the protected network. You should be sure you understand what access lists do before you configure your firewall. For more information about access control lists, refer to the “Access Control Lists: Overview and Guidelines” chapter.

A basic initial configuration allows all network traffic to flow from the protected networks to the unprotected networks, while blocking network traffic from any unprotected networks.

- Permit CBAC traffic to leave the network through the firewall.

All access lists that evaluate traffic leaving the protected network should permit traffic that will be inspected by CBAC. For example, if Telnet will be inspected by CBAC, then Telnet traffic should be permitted on all access lists that apply to traffic leaving the network.

- Use extended access lists to deny CBAC return traffic entering the network through the firewall.

For temporary openings to be created in an access list, the access list must be an extended access list. So wherever you have access lists that will be applied to returning traffic, you must use extended access lists. The access lists should deny CBAC return traffic because CBAC will open up temporary holes in the access lists. (You want traffic to be normally blocked when it enters your network.)

Note If your firewall only has two connections, one to the internal network and one to the external network, using all inbound access lists works well because packets are stopped before they get a chance to affect the router itself.

Basic Configuration

The first time you configure the Cisco IOS Firewall, it is helpful to start with a basic access list configuration that makes the operation of the firewall easy to understand without compromising security. The basic configuration allows all network traffic from the protected networks access to the unprotected networks, while blocking all network traffic (with some exceptions) from the unprotected networks to the protected networks.

Any firewall configuration depends on your site security policy. If the basic configuration does not meet your initial site security requirements, configure the firewall to meet your policy. If you are unfamiliar with that policy or need help with the configuration, contact your network administration group for assistance. For additional guidelines on configuring a firewall, refer to the "Verifying CBAC" section in this chapter.

Use the following guidelines for configuring the initial firewall access lists:

- Do not configure an access list for traffic from the protected networks to the unprotected networks, meaning that all traffic from the protected networks can flow through the interface.

This helps to simplify firewall management by reducing the number of access lists applied at the interfaces. Of course this assumes a high level of trust for the users on the protected networks, and it assumes there are no malicious users on the protected networks who might launch attacks from the "inside." You can fine tune network access for users on the protected networks as you gain experience with access list configuration and the operation of the firewall.

- Configure an access list that includes entries permitting certain ICMP traffic from unprotected networks.

While an access list that denies all IP traffic not part of a connection inspected by CBAC seems most secure, it is not practical for normal operation of the router. The router expects to see ICMP traffic from other routers in the network. Additionally, ICMP traffic is not inspected by CBAC, meaning specific entries are needed in the access list to permit return traffic for ICMP commands. For example, a user on a protected network uses the **ping** command to get the status of a host on an unprotected network; without entries in the access list that permit **echoreply** messages, the user on the protected network gets no response to the **ping** command.

Include access list entries to permit the following ICMP messages:

Message	**Description**
echo reply	Outgoing ping commands require echo-reply messages to come back.
time-exceeded	Outgoing traceroute commands require time-exceeded messages to come back.
packet-too-big	Path MTU discovery requires "too-big" messages to come back.
traceroute	Allow an incoming traceroute.
unreachable	Permit all "unreachable" messages to come back. If a router cannot forward or deliver a datagram, it sends an ICMP unreachable message back to the source and drops the datagram.

- Add an access list entry denying any network traffic from a source address matching an address on the protected network.

This is known as anti-spoofing protection because it prevents traffic from an unprotected network from assuming the identity of a device on the protected network.

- Add an entry denying broadcast messages with a source address of 255.255.255.255.

This entry helps to prevent broadcast attacks.

- By default, the last entry in an extended access list is an implicit denial of all IP traffic not specifically allowed by other entries in the access list.

Although this is the default setting, this final deny statement is not shown by default in an access list. Optionally, you can add an entry to the access list denying IP traffic with any source or destination address with no undesired effects.

For complete information about how to configure IP access lists, refer to the “Configuring IP Services” chapter of the *CiscoIOSIPAddressingServicesConfigurationGuide*.

For tips on applying access lists at an external or internal interface, review the sections "External Interface" and "Internal Interface" in this chapter.

External Interface

Here are some guidelines for your access lists when you will be configuring CBAC on an external interface:

- If you have an outbound IP access list at the external interface, the access list can be a standard or extended access list. This outbound access list should permit traffic that you want to be inspected by CBAC. If traffic is not permitted, it will not be inspected by CBAC, but will be simply dropped.
- The inbound IP access list at the external interface must be an extended access list. This inbound access list should deny traffic that you want to be inspected by CBAC. (CBAC will create temporary openings in this inbound access list as appropriate to permit only return traffic that is part of a valid, existing session.)
- For complete information about how to configure IP access lists, refer to the “Configuring IP Services” chapter of the *CiscoIOSIPAddressingServicesConfigurationGuide*.

Internal Interface

Here are some tips for your access lists when you will be configuring CBAC on an internal interface:

- If you have an inbound IP access list at the internal interface or an outbound IP access list at external interface(s), these access lists can be either a standard or extended access list. These access lists should permit traffic that you want to be inspected by CBAC. If traffic is not permitted, it will not be inspected by CBAC, but will be simply dropped.
- The outbound IP access list at the internal interface and the inbound IP access list at the external interface must be extended access lists. These outbound access lists should deny traffic that you want to be inspected by CBAC. (CBAC will create temporary openings in these outbound access lists as appropriate to permit only return traffic that is part of a valid, existing session.) You do not necessarily need to configure an extended access list at both the outbound internal interface and the inbound external interface, but at least one is necessary to restrict traffic flowing through the firewall into the internal protected network.

For complete information about how to configure IP access lists, refer to the “Configuring IP Services” chapter of the *CiscoIOSIPAddressingServicesConfigurationGuide*.

Half-Open Sessions

An unusually high number of half-open sessions (either absolute or measured as the arrival rate) could indicate that a denial-of-service attack is occurring. For TCP, “half-open” means that the session has not reached the established state--the TCP three-way handshake has not yet been completed. For UDP, “half-open” means that the firewall has detected no return traffic.

CBAC measures both the total number of existing half-open sessions and the rate of session establishment attempts. Both TCP and UDP half-open sessions are counted in the total number and rate measurements. Rate measurements are made several times per minute.

When the number of existing half-open sessions rises above a threshold (the **max-incompletehigh** number), the software will delete half-open sessions as required to accommodate new connection requests. The software will continue to delete half-open requests as necessary, until the number of existing half-open sessions drops below another threshold (the **max-incompletelow** number).

When the rate of new connection attempts rises above a threshold (the **one-minutehigh** number), the software will delete half-open sessions as required to accommodate new connection attempts. The software will continue to delete half-open sessions as necessary, until the rate of new connection attempts drops below another threshold (the **one-minutelow** number). The rate thresholds are measured as the number of new session connection attempts detected in the last one-minute sample period.

IP Packet Fragmentation Inspection

CBAC inspection rules can help protect hosts against certain DoS attacks involving fragmented IP packets.

Using fragmentation inspection, the firewall maintains an interfragment state (structure) for IP traffic. Non-initial fragments are discarded unless the corresponding initial fragment was permitted to pass through the firewall. Non-initial 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. Applying 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 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 disabled by default. Fragmentation detection must be explicitly enabled for an inspection rule using the **ipinspectname** 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, gets some fraction of the firewall's fragment state resources, and legitimate, unfragmented traffic can flow through the firewall unimpeded.

Generic 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 than 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 to the FTP state information.

With TCP and UDP inspection, packets entering the network must exactly match the corresponding packet that previously exited the network. The entering packets must have the same source/destination addresses and source/destination port numbers as the exiting packet (but reversed); otherwise, the entering packets will be blocked at the interface. Also, all TCP packets with a sequence number outside of the window are dropped.

With UDP inspection configured, replies will only be permitted back in through the firewall if they are received within a configurable time after the last request was sent out. (This time is configured with the **ipinspectudpidle-time** command.)

Guidelines for Configuring a Firewall

As with all networking devices, you should always protect access into the firewall by configuring passwords as described in the module “Configuring Security with Passwords, Privilege Levels, and Login Usernames for CLI Sessions on Networking Devices.” You should also consider configuring user authentication, authorization, and accounting as described in the “Authentication, Authorization, and Accounting (AAA)” part of this guide.

You should also consider the following recommendations:

- When setting passwords for privileged access to the firewall, use the **enablesecret** command rather than the **enablepassword** command, which does not have as strong an encryption algorithm.
- Put a password on the console port. In authentication, authorization, and accounting (AAA) environments, use the same authentication for the console as for elsewhere. In a non-AAA environment, at a minimum configure the **login** and **password***password* commands.
- Think about access control before you connect a console port to the network in any way, including attaching a modem to the port. Be aware that a *break* on the console port might give total control of the firewall, even with access control configured.
- Apply access lists and password protection to all virtual terminal ports. Use access lists to limit who can Telnet into your router.
- Do not enable any local service (such as SNMP or NTP) that you do not use. Cisco Discovery Protocol (CDP) and Network Time Protocol (NTP) are on by default, and you should turn these off if you do not need them.

To turn off CDP, enter the **nocdprun** global configuration command. To turn off NTP, enter the **ntpdisable** interface configuration command on each interface not using NTP.

If you must run NTP, configure NTP only on required interfaces, and configure NTP to listen only to certain peers.

Any enabled service could present a potential security risk. A determined, hostile party might be able to find creative ways to misuse the enabled services to access the firewall or the network.

For local services that are enabled, protect against misuse. Protect by configuring the services to communicate only with specific peers, and protect by configuring access lists to deny packets for the services at specific interfaces.

- Protect against spoofing: protect the networks on both sides of the firewall from being spoofed from the other side. You could protect against spoofing by configuring input access lists at all interfaces to pass only traffic from expected source addresses, and to deny all other traffic.

You should also disable source routing. For IP, enter the **noipsource-route** global configuration command. Disabling source routing at all routers can also help prevent spoofing.

You should also disable minor services. For IP, enter the **noservicetcp-small-servers** and **noserviceudp-small-servers** global configuration commands. In Cisco IOS Release 12.0 and later, these services are disabled by default.

- Prevent the firewall from being used as a relay by configuring access lists on any asynchronous Telnet ports.
- Normally, you should disable directed broadcasts for all applicable protocols on your firewall and on all your other routers. For IP, use the **noipdirected-broadcast** command. Rarely, some IP networks do require directed broadcasts; if this is the case, do not disable directed broadcasts.

Directed broadcasts can be misused to multiply the power of denial-of-service attacks, because every denial-of-service packet sent is broadcast to every host on a subnet. Furthermore, some hosts have other intrinsic security risks present when handling broadcasts.

- Configure the **noproxy-arp** command to prevent internal addresses from being revealed. (This is important to do if you do not already have NAT configured to prevent internal addresses from being revealed.)
- Keep the firewall in a secured (locked) room.

RTSP Inspection

-

RTSP with RDT

The following example illustrates the result of the **showipinspectsession** command. It shows that a control channel (rtsp) and data channel (rtsp-data) are open between hosts 192.168.155.2 and 192.168.35.1.

```
router# show ip inspect session

Established Sessions
 Session 616B4F1C (192.168.155.2:7548)=>(192.168.35.1:6970) rtsp-data SIS_OPEN
 Session 611E2904 (192.168.35.1:1221)=>(192.168.155.2:554) rtsp SIS_OPEN
```

The following example illustrates the result of the **showipaccess-list** command. It shows that two dynamic entries (permit statements) were added to ACL 100 for the multimedia session. The TCP entry creates a dynamic opening through the firewall between port 554 (RTSP protocol port) on the client and port 1221 on the server. The UDP entry creates a dynamic opening between data port 7548 on the client and data port 6970 on the server.

```
router# show ip access-list
Extended IP access list 100
 permit udp host 192.168.155.2 eq 7548 host 192.168.35.1 eq 6970 (31 matches)
 permit tcp host 192.168.155.2 eq 554 host 192.168.35.1 eq 1221 (27 matches)
```

After closing the multimedia session, review the session output using the **show** commands to verify the firewall software has removed the dynamic entries from the configuration.

RTSP with TCP Only Interleaved Mode

The following example illustrates the result of the **showipinspectsession** command. It shows that only a single control channel (rtsp) is open between hosts 192.168.155.2 and 192.168.35.1. In this mode, data is tunneled through the firewall using the TCP connection to interleave RDT or RTP data.

```
router# show ip inspect session
```

```
Established Sessions
 Session 611E2904 (192.168.35.1:1228)=>(192.168.155.2:554) rtsp SIS_OPEN
```

The following example illustrates the result of the **showipaccess-list** command. It shows that a single dynamic entry (permit statement) was added to ACL 100 for the multimedia session. The TCP entry creates a dynamic opening through the firewall between port 554 (RTSP protocol port) on the client and port 1228 on the server.

```
router# show ip access-lists

Extended IP access list 100
 permit tcp host 192.168.155.2 eq 554 host 192.168.35.1 eq 1228 (391 matches)
```

After closing the multimedia session, review the session output using the **show** commands to verify the firewall software has removed the dynamic entries from the configuration.

RTSP with SMIL

The following example illustrates the result of the **showipinspectsession** command for RTSP using Synchronized Multimedia Integration Language (SMIL). It shows that a single control channel (rtsp) and multiple data channels (rtsp-data) are open between hosts 192.168.155.2 and 192.168.35.1. The data channels appear as half open sessions because the UDP data flows in one direction only, which is from the server to the client.

```
router# show ip inspect session

Established Sessions
 Session 616CA914 (192.168.155.2:30616)=>(192.168.35.1:6974) rtsp-data SIS_OPEN
 Session 616B4E78 (192.168.35.1:1230)=>(192.168.155.2:554) rtsp SIS_OPEN
 Session 614AB61C (192.168.155.2:29704)=>(192.168.35.1:6976) rtsp-data SIS_OPEN
 Session 616CAA88 (192.168.155.2:26764)=>(192.168.35.1:6972) rtsp-data SIS_OPEN
Half-open Sessions
 Session 614AAEF0 (192.168.155.2:15520)=>(192.168.35.1:6970) rtsp-data SIS_OPENING
```

The following example illustrates the result of the **showipaccess-lists** command. It shows that multiple dynamic entries (permit statements) were added to ACL 100 for the multimedia session. The TCP entry creates a dynamic opening through the firewall between port 554 (RTSP protocol port) on the client and port 1230 on the server. The UDP entries create dynamic openings between negotiated data ports on the client (192.168.155.2) and the server (192.168.35.1).

```
router# show ip access-list

Extended IP access list 100
 permit udp host 192.168.155.2 eq 29704 host 192.168.35.1 eq 6976 (182 matches)
 permit udp host 192.168.155.2 eq 30616 host 192.168.35.1 eq 6974 (268 matches)
 permit udp host 192.168.155.2 eq 26764 host 192.168.35.1 eq 6972 (4 matches)
 permit udp host 192.168.155.2 eq 15520 host 192.168.35.1 eq 6970 (12 matches)
 permit tcp host 192.168.155.2 eq 554 host 192.168.35.1 eq 1230 (41 matches)
```

After closing the multimedia session, review the session output using the **show** commands to verify the firewall software has removed the dynamic entries from the configuration.

RTSP with RTP IP TV

The following example illustrates the result of the **showipinspectsession** command for RTSP with the Cisco IP/TV application. The output shows that a single control channel (rtsp) and multiple data channels (rtsp-data) are open between hosts 192.168.2.15 and 192.168.102.23. The data channels appear as half-open sessions because the UDP data flows in one direction only, which is from the server to the client.

```
router# show ip inspect session

Established Sessions
```

```
 Session 611493C0 (192.168.2.15:2571)=>(192.168.102.23:8554) rtsp SIS_OPEN
Half-open Sessions
 Session 6114A22C (192.168.102.23:2428)=>(192.168.2.15:20112) rtsp-data SIS_OPENING
 Session 61149F44 (192.168.102.23:2428)=>(192.168.2.15:20113) rtsp-data SIS_OPENING
 Session 6114A0B8 (192.168.102.23:2429)=>(192.168.2.15:20115) rtsp-data SIS_OPENING
 Session 6114A3A0 (192.168.102.23:2429)=>(192.168.2.15:20114) rtsp-data SIS_OPENING
```

The following example illustrates the result of the **showipaccess-lists** command. It shows that multiple dynamic entries (permit statements) were added to ACL 100 for the multimedia session. The TCP entry creates a dynamic opening through the firewall between port 554 (RTSP protocol port) on the client and port 1230 on the server. The UDP entries create dynamic openings between negotiated data ports on the client (192.168.2.15) and the server (192.168.102.23).

```
router# show ip access-lists

Extended IP access list 100
 permit udp host 192.168.102.23 eq 2428 host 192.168.2.15 eq 20113 (11 matches)
 permit udp host 192.168.102.23 eq 2428 host 192.168.2.15 eq 20112 (256 matches)
 permit udp host 192.168.102.23 eq 2429 host 192.168.2.15 eq 20115 (11 matches)
 permit udp host 192.168.102.23 eq 2429 host 192.168.2.15 eq 20114 (4598 matches)
 permit tcp host 192.168.102.23 eq 8554 host 192.168.2.15 eq 2571 (22 matches)
```

After closing the multimedia session, review the session output using the **show** commands to verify that the firewall software has removed the dynamic entries from the configuration.

H.323 V2

The following example illustrates the result of the **showipinspectsession** command for H.323 V2. It shows a single H.323 control channel, an RTP Control Protocol channel for both audio and video data, and an RTP data channel between hosts 192.168.155.2 and 192.168.35.1.

```
Session 615E2688 (192.168.35.1:49609)=>(192.168.155.1:49609) H323-RTCP-audio SIS_OPEN
Session 615E2688 (192.168.35.1:49508)=>(192.168.155.1:49508) H323-RTP-audio SIS_OPEN
Session 615E2688 (192.168.35.1:49410)=>(192.168.155.1:49410) H323-RTP-video SIS_OPEN
Session 615E2688 (192.168.35.1:49611)=>(192.168.155.1:49611) H323-RTCP-video SIS_OPEN
Session 615E1640 (192.168.35.1:4414)=>(192.168.155.1:1720) H323 SIS_OPEN
```

The following example illustrates the result of the **showipaccess-lists** command. It shows that multiple dynamic entries (permit statements) were added to ACL 100 for the multimedia session. The TCP entry creates a dynamic opening through the firewall between port 1720 (H.323 V2 protocol port) on the client and port 4414 on the server. The UDP entries create dynamic openings between negotiated data ports on the client (192.168.155.1) and the server (192.168.35.1).

```
Router# show ip access-lists

Extended IP access list 100
 permit udp host 192.168.155.1 eq 49609 host 192.168.35.1 eq 49609 (11 matches)
 permit udp host 192.168.155.1 eq 49508 host 192.168.35.1 eq 49508 (256 matches)
 permit udp host 192.168.155.1 eq 49411 host 192.168.35.1 eq 49411 (11 matches)
 permit udp host 192.168.155.1 eq 49610 host 192.168.35.1 eq 49610 (4598 matches)
 permit tcp host 192.168.155.1 eq 1720 host 192.168.35.1 eq 4414 (22 matches)
```

Interpreting Syslog and Console Messages Generated by CBAC

CBAC provides syslog messages, console alert messages, and audit trail messages. These messages are useful because they can alert you to network attacks and because they provide an audit trail that provides details about sessions inspected by CBAC. While they are generally referred to as error messages, not all error messages indicate problems with your system.

Audit trail and alert information is configurable on a per-application basis using the CBAC inspection rules.

For explanations and recommended actions related to the error messages mentioned in this section, refer to the *Cisco IOS System Error Messages*.

Denial-of-Service Attack Detection Error Messages

CBAC detects and blocks denial-of-service attacks and notifies you when denial-of-service attacks occur. Error messages such as the following may indicate that denial-of-service attacks have occurred:

```
%FW-4-ALERT_ON: getting aggressive, count (550/500) current 1-min rate: 250
%FW-4-ALERT_OFF: calming down, count (0/400) current 1-min rate: 0
```

When %FW-4-ALERT_ON and %FW-4-ALERT_OFF error messages appear together, each "aggressive/calming" pair of messages indicates a separate attack. The preceding example shows one separate attack.

Error messages such as the following may indicate that a denial-of-service attack has occurred on a specific TCP host:

```
%FW-4-HOST_TCP_ALERT_ON: Max tcp half-open connections (50) exceeded for host 172.21.127.242.
%FW-4-BLOCK_HOST: Blocking new TCP connections to host 172.21.127.242 for 2 minutes (half-open
 count 50 exceeded)
%FW-4-UNBLOCK_HOST: New TCP connections to host 172.21.127.242 no longer blocked
```

SMTP Attack Detection Error Messages

CBAC detects and blocks SMTP attacks (illegal SMTP commands) and notifies you when SMTP attacks occur. Error messages such as the following may indicate that an SMTP attack has occurred:

```
%FW-4-SMTP_INVALID_COMMAND: Invalid SMTP command from initiator (192.168.12.3:52419)
```

CBAC also detects a limited number of SMTP attack signatures. A signature in a SYSLOG message indicates a possible attack against the protected network, such as the detection of illegal SMTP commands in a packet. Whenever a signature is detected, the connection will be reset.

The Cisco IOS Firewall supports the following SMTP attack signatures:

Signature	Description
Mail: bad rcpt	Triggers on any mail message with a "pipe" (|) symbol in the recipient field.
Mail: bad from	Triggers on any mail message with a "pipe" (|) symbol in the "From:" field.
Mail: old attack	Triggers when "wiz" or "debug" commands are sent to the SMTP port.
Mail: decode	Triggers on any mail message with a ":decode@" in the header.
Majordomo	A bug in the Majordomo program will allow remote users to execute arbitrary commands at the privilege level of the server.

The following is a sample SMTP attack signature message:

```
02:04:55: %FW-4-TCP_MAJORDOMO_EXEC_BUG: Sig:3107:Majordomo Execute Attack - from 192.168.25.1
 to 192.168.205.1:
```

Java Blocking Error Messages

CBAC detects and selectively blocks Java applets and notifies you when a Java applet has been blocked. Error messages such as the following may indicate that a Java applet has been blocked:

```
%FW-4-HTTP_JAVA_BLOCK: JAVA applet is blocked from (172.21.127.218:80) to
(172.16.57.30:44673).
```

FTP Error Messages

CBAC detects and prevents certain FTP attacks and notifies you when this occurs. Error messages such as the following may appear when CBAC detects these FTP attacks:

```
%FW-3-FTP_PRIV_PORT: Privileged port 1000 used in PORT command -- FTP client 10.0.0.1  FTP
 server 10.1.0.1
%FW-3-FTP_SESSION_NOT_AUTHENTICATED: Command issued before the session is authenticated
-- FTP client 10.0.0.1
%FW-3-FTP_NON_MATCHING_IP_ADDR: Non-matching address 172.19.148.154 used in PORT command
-- FTP client 172.19.54.143  FTP server 172.16.127.242
```

Audit Trail Messages

CBAC provides audit trail messages to record details about inspected sessions. Audit trail information is configurable on a per-application basis using the CBAC inspection rules. To determine which protocol was inspected, use the responder’s port number. The port number follows the responder’s address. The following are sample audit trail messages:

```
%FW-6-SESS_AUDIT_TRAIL: tcp session initiator (192.168.1.13:33192) sent 22 bytes -- responder
 (192.168.129.11:25) sent 208 bytes
%FW-6-SESS_AUDIT_TRAIL: http session initiator (172.16.57.30:44673) sent 1599 bytes --
responder (172.21.127.218:80) sent 93124 bytes
```

Turning Off CBAC

You can turn off CBAC using the **noipinspect** global configuration command.

The **noipinspect** command removes all CBAC configuration entries and resets all CBAC global timeouts and thresholds to the defaults. All existing sessions are deleted and their associated access lists removed.

In most situations, turning off CBAC has no negative security impact because CBAC creates “permit” access lists. Without CBAC configured, no “permit” access lists are maintained. Therefore, no derived traffic (returning traffic or traffic from the data channels) can go through the firewall. The exception is SMTP and Java blocking. With CBAC turned off, unacceptable SMTP commands or Java applets may go through the firewall.

How to Configure Context-Based Access Control

Configuring Global Timeouts and Thresholds

CBAC uses timeouts and thresholds to determine how long to manage state information for a session, and to determine when to drop sessions that do not become fully established. These timeouts and thresholds apply globally to all sessions.

You can use the default timeout and threshold values, or you can change to values more suitable to your security requirements. You should make any changes to the timeout and threshold values before you continue configuring CBAC.

To reset any threshold or timeout to the default value, use the **no** form of the command in the table below.

Note If you want to enable the more aggressive TCP host-specific denial-of-service prevention that includes the blocking of connection initiation to a host, you must set the **block-time** specified in the **ipinspecttcpmax-incompletehost** command (see the last row in the table below).

All the available CBAC timeouts and thresholds are listed in the table below, along with the corresponding command and default value. To change a global timeout or threshold listed in the "Timeout of Threshold Value to Change" column, use the global configuration command in the "Command" column:

Table 1: Timeout and Threshold Values

Timeout or Threshold Value to Change	Command	Default
The length of time the software waits for a TCP session to reach the established state before dropping the session.	**ip inspect tcp synwait-time** *seconds*	30 seconds
Disable the window scale option check for a TCP packet that has an invalid window scale option under the Context-Based Access Control (CBAC) firewall.	**ip inspect tcp window-scale-enforcement loose**	The strict window scale option check is enabled in the firewall by default.
The length of time a TCP session will still be managed after the firewall detects a FIN-exchange.	**ip inspect tcp finwait-time** *seconds*	5 seconds
The length of time a TCP session will still be managed after no activity (the TCP idle timeout).[1]	**ip inspect tcp idle-time** *seconds*	3600 seconds (1 hour)
The length of time a UDP session will still be managed after no activity (the UDP idle timeout). 1	**ip inspect udp idle-time** *seconds*	30 seconds
The length of time a DNS name lookup session will still be managed after no activity.	**ip inspect dns-timeout** *seconds*	5 seconds
The number of existing half-open sessions that will cause the software to start deleting half-open sessions.[2]	**ip inspect max-incomplete high** *number*	500 existing half-open sessions
		400 existing half-open sessions

Timeout or Threshold Value to Change	Command	Default
The number of existing half-open sessions that will cause the software to stop deleting half-open sessions. 2	**ip inspect max-incomplete low** *number*	
The rate of new sessions that will cause the software to start deleting half-open sessions. 2	**ip inspect one-minute high** *number*	500 half-open sessions per minute
The rate of new sessions that will cause the software to stop deleting half-open sessions. 2	**ip inspect one-minute low** *number*	400 half-open sessions per minute
The number of existing half-open TCP sessions with the same destination host address that will cause the software to start dropping half-open sessions to the same destination host address.[3]	**ip inspect tcp max-incomplete host** *number* **block-time** *minutes*	50 existing half-open TCP sessions; 0 minutes

[1] The global TCP and UDP idle timeouts can be overridden for specified application-layer protocols' sessions as described in the ip inspect name (global configuration) command description, found in the "Context-Based Access Control Commands" chapter of the Cisco IOS Security Command Reference.

[2] See the following section, "Half-Open Sessions," for more information.

[3] Whenever the max-incomplete host threshold is exceeded, the software will drop half-open sessions differently depending on whether the block-time timeout is zero or a positive non-zero number. If the block-time timeout is zero, the software will delete the oldest existing half-open session for the host for every new connection request to the host and will let the SYN packet through. If the block-time timeout is greater than zero, the software will delete all existing half-open sessions for the host, and then block all new connection requests to the host. The software will continue to block all new connection requests until the block-time expires.

Defining an Inspection Rule

After you configure global timeouts and thresholds, you must define an inspection rule. This rule specifies what IP traffic (which application-layer protocols) will be inspected by CBAC at an interface.

Normally, you define only one inspection rule. The only exception might occur if you want to enable CBAC in two directions as described earlier in the section "When and Where to Configure CBAC." For CBAC configured in both directions at a single firewall interface, you should configure two rules, one for each direction.

An inspection rule should specify each desired application-layer protocol as well as generic TCP or generic UDP if desired. The inspection rule consists of a series of statements each listing a protocol and specifying the same inspection rule name.

Inspection rules include options for controlling alert and audit trail messages and for checking IP packet fragmentation.

Configuring Application-Layer Protocol Inspection

Note For CBAC inspection to work with NetMeeting 2.0 traffic (an H.323 application-layer protocol), you must also configure inspection for TCP, as described later in the "Configuring Generic TCP and UDP Inspection" section. This requirement exists because NetMeeting 2.0 uses an additional TCP channel not defined in the H.323 specification.

Configuring Application-Layer Protocols

To configure CBAC inspection for an application-layer protocol, use one or both of the following commands in global configuration mode:

Command	Purpose
`Router(config)#` **ip inspect name** *inspection-name* *protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]	Configures CBAC inspection for an application-layer protocol (except for RPC and Java). Use one of the protocol keywords defined in the table above. Repeat this command for each desired protocol. Use the same *inspection-name* value to create a single inspection rule.
`Router(config)#` **ip inspect name** *inspection-name* **rpc program-number** *number* [**wait-time** *minutes*] [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]	Enables CBAC inspection for the RPC application-layer protocol. You can specify multiple RPC program numbers by repeating this command for each program number. Use the same *inspection-name* value to create a single inspection rule.

Configuring Java Blocking

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 external sites except for those you specifically designate as hostile.)

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 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 from FTP, gopher, HTTP on a nonstandard port, and so forth.

To block all Java applets except for applets from friendly locations, use the following commands in global configuration mode:

SUMMARY STEPS

1. Do one of the following:
   - Router(config)# **ipaccess-liststandard***name* **permit** ... **deny** ... (Use permit and deny statements as appropriate.)
   -
   -
   - Router(config)# **access-list***access-list-number* {**deny** | **permit**} *protocolsource* [*source-wildcard*]eq www *destination* [*destination-wildcard*]
2. Router(config)# **ipinspectnameinspection-namehttp**[**java-list***access-list*] [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout***seconds*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Do one of the following: • Router(config)# **ipaccess-liststandard***name* **permit** ... **deny** ... (Use permit and deny statements as appropriate.) • •	Creates a standard access list that permits traffic only from friendly sites, and denies traffic from hostile sites. Use the **any** keyword for the destination as appropriate--but be careful to not misuse the **any** keyword to inadvertently allow all applets through.

	Command or Action	Purpose
	• Router(config)# **access-list***access-list-number* {**deny** | **permit**} *protocolsource* [*source-wildcard*]eq www *destination* [*destination-wildcard*]	
Step 2	Router(config)# **ipinspectnameinspection-namehttp**[**java-list***access-list*] [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout***seconds*]	Blocks all Java applets except for applets from the friendly sites defined previously in the access list. Java blocking only works with numbered standard access lists. To create a single inspection rule, use the same*inspection-name* value as when you specified other protocols.

Configuring Generic TCP and UDP Inspection

To configure CBAC inspection for TCP or UDP packets, use one or both of the following commands in global configuration mode:

Command	Purpose
Router(config)# **ip inspect name** *inspection-name* **tcp** [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]	Enables CBAC inspection for TCP packets. To create a single inspection rule, use the same*inspection-name* value as when you specified other protocols.
Router(config)# **ip inspect name** *inspection-name* **udp** [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]	Enables CBAC inspection for UDP packets. To create a single inspection rule, use the same *inspection-name* value as when you specified other protocols.

Applying the Inspection Rule to an Interface

After you define an inspection rule, you apply this rule to an interface.

Normally, you apply only one inspection rule to one interface. The only exception might occur if you want to enable CBAC in two directions as described earlier in the section “When and Where to Configure CBAC.” For CBAC configured in both directions at a single firewall interface, you should apply two rules, one for each direction.

If you are configuring CBAC on an external interface, apply the rule to outbound traffic.

If you are configuring CBAC on an internal interface, apply the rule to inbound traffic.

To apply an inspection rule to an interface, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **ip inspect** *inspection-name* {**in** | **out**}	Applies an inspection rule to an interface.

Configuring Logging and Audit Trail

Turn on logging and audit trail to provide a record of network access through the firewall, including illegitimate access attempts, and inbound and outbound services. To configure logging and audit trail functions, enter the following commands in global configuration mode:

SUMMARY STEPS

1. Router(config)# **servicetimestampslogdatetime**
2. Router(config)# **logging***host*
3. Router(config)# **loggingfacility***facility-type*
4. Router(config)# **loggingtrap***level*
5. Router(config)#**ipinspectaudit-trail**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **servicetimestampslogdatetime**	Adds the date and time to syslog and audit trail messages.
Step 2	Router(config)# **logging***host*	Specifies the host name or IP address of the host where you want to send syslog messages.
Step 3	Router(config)# **loggingfacility***facility-type*	Configures the syslog facility in which error messages are sent.
Step 4	Router(config)# **loggingtrap***level*	(Optional) Uses this command to limit messages logged to the syslog servers based on severity. The default is level 7 (informational).
Step 5	Router(config)#**ipinspectaudit-trail**	Turns on CBAC audit trail messages.

Verifying CBAC

In most cases, you can tell whether CBAC is inspecting network traffic properly because network applications are working as expected. In some cases, however, you might want to verify CBAC operation. For example, to verify RTSP or H.323 inspection, initiate an RTSP- or H.323-based application through the firewall. Use the **showipinspectsession** and **showipaccesslists** commands to verify CBAC operation. These commands display the dynamic ACL entries and the established connections for a multimedia session.

You can view and verify CBAC configuration, status, statistics, and session information by using one or more of the following commands in EXEC mode:

Command	Purpose
`Router#` **show ip access-lists**	Displays the contents of all current IP access lists.
`Router#` **show ip inspect name** *inspection-name*	Shows a particular configured inspection rule.
`Router#` **show ip inspect config**	Shows the complete CBAC inspection configuration.
`Router#` **show ip inspect interfaces**	Shows interface configuration with regards to applied inspection rules and access lists.
`Router#` **show ip inspect session** [**detail**]	Shows existing sessions that are currently being tracked and inspected by CBAC.
`Router#` **show ip inspect all**	Shows all CBAC configuration and all existing sessions that are currently being tracked and inspected by CBAC.

Monitoring and Maintaining CBAC

You can watch for network attacks and investigate network problems using debug commands and system messages.

Debugging Context-Based Access Control

To assist CBAC debugging, you can turn on audit trail messages that will be displayed on the console after each CBAC session closes. Audit trail information is also configurable on a per-application basis using the CBAC inspection rules.

Note Effective with Cisco IOS Release 12.4(20)T, the **debugipinspect** command is replaced by the**debugpolicy-firewall** command. See the Cisco IOS Debug Command Reference for more information.

To turn on audit trail messages, use the following command in global configuration mode:

Command	Purpose
`Router(config)#` **ip inspect audit-trail**	Turns on CBAC audit trail messages.

Generic Debug Commands

You can use the following generic **debug** commands, entered in privileged EXEC mode:

Command	Purpose
`Router#` **debug ip inspect function-trace**	Displays messages about software functions called by CBAC.
`Router#` **debug ip inspect object-creation**	Displays messages about software objects being created by CBAC. Object creation corresponds to the beginning of CBAC-inspected sessions.
`Router#` **debug ip inspect object-deletion**	Displays messages about software objects being deleted by CBAC. Object deletion corresponds to the closing of CBAC-inspected sessions.
`Router#` **debug ip inspect events**	Displays messages about CBAC software events, including information about CBAC packet processing.
`Router#` **debug ip inspect timers**	Displays messages about CBAC timer events such as when a CBAC idle timeout is reached.
`Router#` **debug ip inspect detail**	Enables the detailed option, which can be used in combination with other options to get additional information.

Transport Level Debug Commands

You can use the following transport-level **debug** commands, entered in privileged EXEC mode:

Command	Purpose
`Router#` **debug ip inspect tcp**	Displays messages about CBAC-inspected TCP events, including details about TCP packets.
`Router#` **debug ip inspect udp**	Displays messages about CBAC-inspected UDP events, including details about UDP packets.

Application Protocol Debug Commands

You can use the following application protocol **debug** command, entered in privileged EXEC mode:

Command	Purpose
`Router#` **debug ip inspect** *protocol*	Displays messages about CBAC-inspected protocol events, including details about the protocol's packets. Refer to the table to determine the protocol keyword.

CBAC Configuration Examples

The first example develops a CBAC inspection rule for specific protocols and a supporting access control list (ACL). This example focuses how to configure CBAC; it does not provide a complete router configuration and does not describe other elements of the configuration.

The next example develops a CBAC inspection rule for sites that might have remote traffic through an ATM interface. This example further illustrates on how to configure CBAC and emphasizes the application of the configuration rule at the interface, whatever that interface might be. This example does not provide a complete router configuration and does not describe other elements of the configuration.

The remote-office examples also focus on the firewall configuration but do not provide detailed descriptions of other configuration elements, such as the Basic Rate Interface (BRI) and dialer interface configurations.

Other examples provide more complete firewall configurations, further illustrating ways in which to apply CBAC.

In each example, configuring protocol inspection using CBAC has four components:

- Defining an access list with the appropriate permissions.
- Applying the ACL at an interface where you want to control access.
- Defining an inspection rule that includes the protocol that you want to inspect.
- Applying the inspection rule at an interface where you want to inspect traffic.

Ethernet Interface Configuration Example

This example looks at each of these four components. For this example, CBAC is being configured to inspect RTSP and H.323 protocol traffic inbound from the protected network on a router with two Ethernet interfaces. Interface Ethernet1/0 is the protected network and interface Ethernet1/1 is the unprotected network. The security policy for the protected site uses access control lists (ACLs) to restrict inbound traffic on the unprotected interface to specific ICMP protocol traffic, denying inbound access for TCP and UDP protocol traffic. Inbound access for specific protocol traffic is provided through dynamic access lists, which are generated according to CBAC inspection rules.

ACL 100 denies TCP and UDP traffic from any source or destination while permitting specific ICMP protocol traffic. The final deny statement is not required, but is included for explicitness--the final entry in any ACL is an implicit denial of all IP protocol traffic.

```
Router(config)# access-list 100 deny tcp any any

Router(config)# access-list 100 deny udp any any

Router(config)# access-list 100 permit icmp any any echo-reply

Router(config)# access-list 100 permit icmp any any time-exceeded

Router(config)# access-list 100 permit icmp any any packet-too-big

Router(config)# access-list 100 permit icmp any any traceroute

Router(config)# access-list 100 permit icmp any any unreachable

Router(config)# access-list 100 deny ip any any
```

ACL 100 is applied inbound at interface Ethernet1/1 to block all access from the unprotected network to the protected network.

```
Router(config)# interface Ethernet1/1

Router(config-if)# ip access-group 100 in
```

An inspection rule is created for “hqusers” that covers two protocols: RTSP and H.323.

```
Router(config)# ip inspect name hqusers rtsp

Router(config)# ip inspect name hqusers h323
```

The inspection rule is applied inbound at interface Ethernet1/0 to inspect traffic from users on the protected network. When CBAC detects multimedia traffic from the protected network, CBAC creates dynamic entries in access list 100 to allow return traffic for multimedia sessions.

```
Router(config)# interface Ethernet1/0

Router(config-if)# ip inspect hqusers in
```

ATM Interface Configuration Example

In this example, CBAC inspection (firewall protection) is required against inbound traffic on an ATM interface. This example might apply to sites where local hosts require access to hosts or services on a remote network. The security policy for this site uses access control lists (ACLs) to restrict inbound traffic on the ATM interface to specific ICMP protocol traffic, denying inbound access for TCP and UDP protocol traffic. Inbound access for specific TCP and UDP protocol traffic is provided through dynamic access lists, which are generated according to CBAC inspection rules.

For information on how to select the interface on which to apply CBAC, refer to the "Configuring IP Access Lists at the Interface" section.

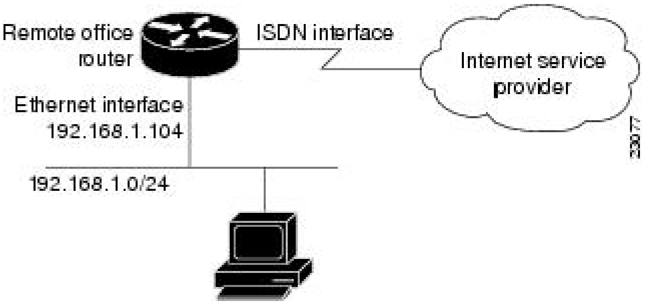
Note For Frame Relay or ATM interfaces, you can apply CBAC inspection rules separately on each sub-interface, even though the subinterfaces are physically connected through one interface.

```
! ------------------------
! Create the Inspection Rule
! -------------------------
!
! Create the CBAC inspection rule “test”, allowing inspection of the protocol traffic
! specified by the rule. This inspection rule sets the timeout value to 30 seconds for
! each protocol (except for RPC). The timeout value defines the maximum time that a
! connection for a given protocol can remain active without any traffic passing through
! the router. When these timeouts are reached, the dynamic ACLs that are inserted to
! permit the returning traffic are removed, and subsequent packets (possibly even valid
! ones) are not permitted.
ip inspect name test cuseeme timeout 30
ip inspect name test ftp timeout 30
ip inspect name test h323 timeout 30
ip inspect name test realaudio timeout 30
ip inspect name test rpc program-number 100000
ip inspect name test streamworks timeout 30
ip inspect name test vdolive timeout 30
!
! ------------------------------
! Create the Access Control List
! ------------------------------
!
! In this example, ACL 105 denies all TCP and UDP protocol traffic. ICMP traffic from
! subnet 192.168.1.0 is permitted to allow access for routing and control traffic.
! ACL 105 specifies that only the return traffic for protocols defined in the
! inspection rule is allow access through the interface where this rule is applied. The
! final deny statement is added for explicitness.
access-list 105 deny TCP any any
access-list 105 deny UDP any any
access-list 105 permit icmp any any echo-reply
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 time-exceeded
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 packet-too-big
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 traceroute
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 unreachable
access-list 105 deny ip any any
!
! ---------------------------------
! Apply the Inspection Rule and ACL
! ---------------------------------
!
! In this example, the inspection rule “test” is applied to traffic at interface ATM3/0
! for connections initiated in the outbound direction; that is, from hosts that are
! located on a local network. CBAC creates dynamic access list entries for traffic
! initiated by local hosts. These dynamic entries allow inbound (returning) traffic for
! that connection. ACL 105 is applied at interface ATM3/0 in the inbound direction to
! block traffic initiated from hosts on a remote network that is not part of an
! existing connection.
interface ATM3/0
ip address 10.1.10.1 255.0.0.0
ip access-group 105 in
no ip directed-broadcast
ip inspect test out
no shutdown
atm clock INTERNAL
atm pvc 7 7 7 aal5snap
map-group atm
```

Remote Office to ISP Configuration Example

This example describes one possible Cisco IOS Firewall configuration for a remote office router connected to an Internet service provider (ISP). In this configuration, the site security policy allows hosts on the local network to initiate traffic to the ISP while traffic inbound to the router from the ISP is blocked at the ISDN interface. Specific ICMP control message traffic is permitted through the firewall. No mail or Web services are available from the local network. the figure below illustrates this example.

Figure 4: Remote Office to ISP Sample Configuration



The firewall has two interfaces:

- An Ethernet interface connects to the internal protected network.

Interface Ethernet0 has no ACL applied to it, meaning that all traffic initiated on the LAN is allowed access to the ISP. In this configuration example, Network Address Translation (NAT) is not turned on, and the addresses on interface Ethernet0 are reserved IP addresses. In a production environment, addresses on Ethernet0 either must be registered network addresses, or you must turn on NAT to hide these inside addresses from being visible on the Internet.

- An ISDN Basic Rate Interface (BRI) connects the router to the ISP. In this example, a dialer profile is used to control the BRI interface. This means that the ACL and CBAC inspection rules are applied at the dialer interface, not directly at the physical ISDN (BRI) interface using a dialer map.

```
! --------------------------------------
! General Cisco IOS Firewall Guidelines
! --------------------------------------
! The following global configuration entries illustrate good security practices.
enable secret 5 <elided>
no ip source-route
no cdp run
!
! --------------------------------
! Create the CBAC inspection rule
! ------------------------------
! Create the CBAC inspection rule STOP to allow inspection of the protocol traffic
! specified by the rule.
ip inspect name STOP tcp
ip inspect name STOP ftp
ip inspect name STOP smtp
ip inspect name STOP h323
ip inspect name STOP rcmd
!
! ---------------------------------
! Create Access Control List 105
! ---------------------------------
! ACL 105 denies all IP protocol traffic except for specific ICMP control traffic.
! This means that only the return traffic for protocols defined in the
! inspection rule and the specified ICMP traffic is allowed access through the
! interface where this rule is applied.
!
! Deny broadcast messages with a source address of 255.255.255.255; this helps to
! prevent broadcast attacks.
access-list 105 deny ip host 255.255.255.255 any
!
```

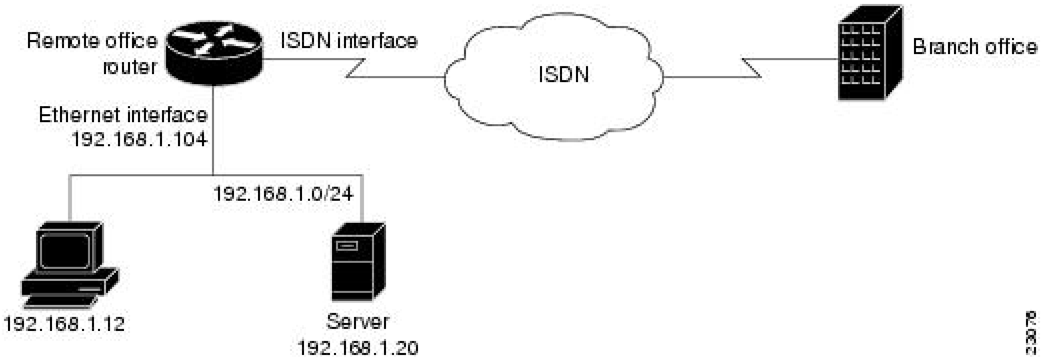
```
! Add anti-spoofing protection by denying traffic with a source address matching a host
! on the Ethernet interface.
acl 105 deny ip 192.168.1.0 0.0.0.255 any
!
! ICMP traffic is not inspected by CBAC. To control the type of ICMP traffic at the
! interface, add static access list entries. This example has the following ICMP
! requirements: outgoing ping commands require echo-reply messages to come back,
! outgoing traceroute commands require time-exceeded messages to come back, path MTU
! discovery requires “too-big” messages to come back, and incoming traceroute
! messages must be allowed. Additionally, permit all “unreachable” messages to come
! back; that is, if a router cannot forward or deliver a datagram, it sends an ICMP
! unreachable message back to the source and drops the datagram.
access-list 105 permit icmp any any echo-reply
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 time-exceeded
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 packet-too-big
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 traceroute
access-list 105 permit icmp any 192.168.1.0 0.0.0.255 unreachable
!
! Final deny for explicitness. This entry is not required but helps complete the access
! list picture. By default, the final entry in any access list is an implicit deny of
! IP protocol traffic. This ensures that the firewall blocks any traffic not explicitly
! permitted by the access list.
access-list 105 deny ip any any
!
! ------------------------------------------------------------
! Configure the interface
! ------------------------------------------------------------
! In this example, no ACLs or inspection rules are applied at interface Ethernet0,
! meaning that all traffic on the local network is allowed to go out. This assumes a
! high-level of trust for the users on the local network.
interface Ethernet0
ip address 192.168.1.104 255.255.255.0
!
no ip directed-broadcast
!
! This example uses a dialer profile, so the ACL and CBAC inspection rules are applied
! at the dialer interface, not the physical BRI interface. The dialer pool-member
! command is used to associate the physical interface with a dialer profile.
interface BRI0
no ip address
no ip directed-broadcast
encapsulation ppp
dialer pool-member 1
isdn switch-type basic-5ess
!
! ---------------------------------------------------------------------
! Create the dialer profile.
! ---------------------------------------------------------------------
! Through the dialer profile, the ACL and CBAC inspection rules are
! applied to every pool member. In this example, the ACL is applied in, meaning that it
! applies to traffic inbound from the ISP. The CBAC inspection rule STOP is applied
! out, meaning that CBAC monitors the traffic through the interface and controls return
! traffic to the router for an existing connection.
interface Dialer0
ip address negotiated
ip access-group 105 in
no ip directed-broadcast
ip inspect STOP out
encapsulation ppp
dialer remote-name <ISP router>
dialer idle-timeout 500
dialer string <elided>
dialer pool 1
dialer-group 1
ppp authentication callin
!
! -----------------------------------------------------------
! Additional entries
! -----------------------------------------------------------
! Configure the router to forward packets destined for an unrecognized subnet of
! a directly connected network.
ip classless
! Route traffic to the dialer interface.
```

```
ip route 0.0.0.0 0.0.0.0 Dialer0
! Include a dialer list protocol entry to specify the protocol that triggers dialing.
dialer-list 1 protocol ip permit
! Add a user name (name of the router your are configuring) and password for caller
! identification and password authentication with the ISP router.
username <router host name> password 5 <elided>
```

Remote Office to Branch Office Configuration Example

This example describes one possible Cisco IOS Firewall configuration for a remote office router connected to a branch office. In this configuration, the site security policy allows hosts on the local network to initiate traffic to the branch office. Mail or Web services are available from a server on the local network, and access to these services is available from the branch office. Traffic from the branch office, except for mail and Web traffic, is blocked at the outside interface. Specific ICMP control message traffic is permitted through the firewall. The figure below illustrates this example.

Figure 5: Remote Office to Branch Office Sample Configuration



The firewall has two interfaces:

- An Ethernet interface connects to the internal protected network.

Interface Ethernet0 has no ACL applied to it, meaning that all traffic initiated from the LAN is allowed access through the firewall.

- An ISDN Basic Rate Interface (BRI) connects the router to the branch office. In this example, a dialer profile is used to control the BRI interface. This means that the ACL and CBAC inspection rules are applied at dialer interface, not directly at the physical ISDN (BRI) interface.

```
! -------------------------------------------
! General firewall configuration guidelines
! -------------------------------------------
! The following global configuration entries illustrate good security practices.
enable secret 5 <elided>
no ip source-route
no cdp run
!
! ---------------------------
! Create the Inspection Rule
! ---------------------------
! Create the CBAC inspection rule STOP to allow inspection of the specified protocol
! traffic. Create the inspection rule GO to allow inspection of SMTP traffic.
ip inspect name STOP tcp
ip inspect name STOP ftp
ip inspect name STOP smtp
ip inspect name STOP h323
ip inspect name GO smtp
!
! ---------------------------------------------------------------
! Create Access Control Lists 106 and 51
! ---------------------------------------------------------------
! ACL 106 permits mail and Web traffic from any host to the specified server. ACL 106
! denies all other ip protocol traffic except for specific ICMP control traffic.
! This means that only the return traffic for protocols defined in the
! inspection rule and the specified ICMP traffic is allowed access through the
! interface where this rule is applied.
!
! Deny broadcast messages with a source address of 255.255.255.255; this helps to
! prevent broadcast attacks.
access-list 106 deny ip host 255.255.255.255 any
!
! Add anti-spoofing protection by denying traffic with a source address matching a host
! on the Ethernet interface.
access-list 106 deny ip 192.168.1.0 0.0.0.255 any
!
! ICMP traffic is not inspected by CBAC. To control the type of ICMP traffic at the
! interface, add static access list entries. This example has the following ICMP
! requirements: outgoing ping commands require echo-reply messages to come back,
! outgoing traceroute commands require time-exceeded messages to come back, path MTU
! discovery requires “too-big” messages to come back, and incoming traceroute must be
! allowed. Additionally, permit all “unreachable” messages to come back; that is, if a
! router cannot forward or deliver a datagram, it sends an ICMP unreachable message
! back to the source and drops the datagram.
access-list 106 permit icmp any any echo-reply
access-list 106 permit icmp any 192.168.1.0 0.0.0.255 time-exceeded
access-list 106 permit icmp any 192.168.1.0 0.0.0.255 packet-too-big
access-list 106 permit icmp any 192.168.1.0 0.0.0.255 traceroute
access-list 106 permit icmp any 192.168.1.0 0.0.0.255 unreachable
!
! Permit mail and Web access to a specific server.
access-list 106 permit tcp any host 192.168.1.20 eq smtp
access-list 106 permit tcp any host 192.168.1.20 eq www
!
! Final deny for explicitness. This entry is not required but helps complete the access
! list picture. By default, the final entry in any access list is an implicit deny of
! IP protocol traffic. This ensures that the firewall blocks any traffic not explicitly
! permitted by the access list.
access-list 106 deny ip any any
!
! -----------------------------------------------------------
! Configure the interface.
! -----------------------------------------------------------
! In this example, no ACLs or inspection rules are applied at interface Ethernet0,
! meaning that all traffic on the local network is allowed to go out. This assumes a
! high-level of trust for the users on the local network.
interface Ethernet0
ip address 192.168.1.104 255.255.255.0
no ip directed-broadcast
!
! This example uses a dialer profile, so the ACL and CBAC inspection rules are applied
! at the dialer interface, not the physical BRI interface. The dialer pool-member
! command is used to associate the physical interface with a dialer profile.
interface BRI0
no ip address
no ip directed-broadcast
encapsulation ppp
dialer pool-member 1
isdn switch-type basic-5ess
!
! -------------------------------------------------------------------
! Apply the ACL and CBAC inspection rules at the dialer interface.
! -------------------------------------------------------------------
! Through the dialer profile, the ACL and CBAC inspection rules are
! applied to every pool member. In this example, the ACL is applied in, meaning that it
! applies to traffic inbound from the branch office. The CBAC inspection rule STOP is
! applied out, meaning that CBAC monitors the traffic and controls return traffic to
! the router for an existing connection. The CBAC inspection rule GO is applied in,
! protecting against certain types of DoS attacks as described in this document. Note
! that the GO inspection rule does not control return traffic because there is no ACL
! blocking traffic in that direction; however, it does monitor the connections.
interface Dialer0
ip address <ISDN interface address>
ip access-group 106 in
no ip directed-broadcast
ip inspect STOP out
ip inspect GO in
encapsulation ppp
dialer remote-name <branch office router>
dialer idle-timeout 500
dialer string <elided>
dialer pool 1
dialer-group 1
ppp authentication
!
! ----------------------------------------------------------
! Additional entries
! ----------------------------------------------------------
! Configure the router to forward packets destined for an unrecognized subnet of
! a directly connected network.
ip classless
! Route traffic to the dialer interface.
ip route 0.0.0.0 0.0.0.0 Dialer0
! Include a dialer list protocol entry to specify the protocol that triggers dialing.
dialer-list 1 protocol ip permit
! Add a user name (name of the router your are configuring) and password for caller
! identification and password authentication with the ISP router.
username <router host name> password 5 <elided>
```

Two-Interface Branch Office Configuration Example

This sample configuration file describes a firewall configured with CBAC. The firewall is positioned between a protected field office's internal network and a WAN connection to the corporate headquarters. CBAC is configured on the firewall in order to protect the internal network from potential network threats coming from the WAN side.

The firewall has two interfaces configured:

- Interface Ethernet0 connects to the internal protected network
- Interface Serial0 connects to the WAN with Frame Relay

```
! -----------------------------------------------------------------------
! This first section contains some configuration that is not required for CBAC,
! but illustrates good security practices. Note that there are no
! services on the Ethernet side. Email is picked up via POP from a server on the
! corporate side.
! -----------------------------------------------------------------------
!
hostname user1-examplecorp-fr
!
boot system flash c1600-fw1600-l
enable secret 5 <elided>
!
username user1 password <elided>
ip subnet-zero
no ip source-route
ip domain-name example.com
ip name-server 172.19.2.132
ip name-server 198.92.30.32
!
!
! -----------------------------------------------------------------------
! The next section includes configuration required specifically for CBAC.
! -----------------------------------------------------------------------
!
! The following commands define the inspection rule “myfw”, allowing
! the specified protocols to be inspected. Note that Java applets will be permitted
! according to access list 51, defined later in this configuration.
ip inspect name myfw cuseeme timeout 3600
ip inspect name myfw ftp timeout 3600
ip inspect name myfw http java-list 51 timeout 30
ip inspect name myfw rcmd timeout 3600
ip inspect name myfw realaudio timeout 3600
ip inspect name myfw smtp timeout 3600
ip inspect name myfw tftp timeout 30
ip inspect name myfw udp timeout 15
ip inspect name myfw tcp timeout 3600
!
! The following interface configuration applies the “myfw” inspection rule to
! inbound traffic at Ethernet 0. Since this interface is on the internal network
! side of the firewall, traffic entering Ethernet 0 is actually
! exiting the internal network. Applying the inspection rule to this interface causes
! inbound traffic (which is exiting the network) to be inspected; return traffic will
! only be permitted back through the firewall if part of a session which began from
! within the network.
! Also note that access list 101 is applied to inbound traffic at Ethernet 0.
! (Traffic blocked by the access list will not be inspected.)
interface Ethernet0
description ExampleCorp Ethernet chez user1
ip address 172.19.139.1 255.255.255.248
ip broadcast-address 172.19.131.7
no ip directed-broadcast
no ip proxy-arp
ip inspect myfw in
ip access-group 101 in
no cdp enable
!
interface Serial0
description Frame Relay (Telco ID 22RTQQ062438-001) to ExampleCorp HQ
no ip address
ip broadcast-address 0.0.0.0
encapsulation frame-relay IETF
no arp frame-relay
bandwidth 56
service-module 56k clock source line
service-module 56k network-type dds
frame-relay lmi-type ansi
!
! Note that the following interface configuration applies access list 111 to
! inbound traffic at the external serial interface. (Inbound traffic is
! entering the network.) When CBAC inspection occurs on traffic exiting the
! network, temporary openings will be added to access list 111 to allow returning
! traffic that is part of existing sessions.
!
interface Serial0.1 point-to-point
ip unnumbered Ethernet0
ip access-group 111 in
bandwidth 56
no cdp enable
frame-relay interface-dlci 16
!
ip classless
ip route 0.0.0.0 0.0.0.0 Serial0.1
!
! The following access list defines “friendly” and “hostile” sites for Java
! applet blocking. Because Java applet blocking is defined in the inspection
! rule “myfw” and references access list 51, applets will be actively denied
! if they are from any of the “deny” addresses and allowed only if they are from
! either of the two “permit” networks.
!
access-list 51 deny   172.19.1.203
access-list 51 deny   172.19.2.147
access-list 51 permit 172.18.0.0 0.1.255.255
access-list 51 permit 192.168.1.0 0.0.0.255
access-list 51 deny   any
```

```
!
! The following access list 101 is applied to interface Ethernet 0 above.
! This access list permits all traffic that should be CBAC inspected, and also
! provides anti-spoofing. The access list is deliberately set up to deny unknown
! IP protocols, because no such unknown protocols will be in legitimate use.
!
access-list 101 permit tcp 172.19.139.0 0.0.0.7 any
access-list 101 permit udp 172.19.139.0 0.0.0.7 any
access-list 101 permit icmp 172.19.139.0 0.0.0.7 any
access-list 101 deny   ip any any
!
! The following access list 111 is applied to interface Serial 0.1 above.
! This access list filters traffic coming in from the external side. When
! CBAC inspection occurs, temporary openings will be added to the beginning of
! this access list to allow return traffic back into the internal network.
! This access list should restrict traffic that will be inspected by
! CBAC. (Remember that CBAC will open holes as necessary to permit returning traffic.)
! Comments precede each access list entry. These entries are not all specifically
! related to CBAC, but are created to provide general good security.
!
! Anti-spoofing.
access-list 111 deny   ip 172.19.139.0 0.0.0.7 any
! Sometimes EIGRP is run on the Frame Relay link. When you use an
! input access list, you have to explicitly allow even control traffic.
! This could be more restrictive, but there would have to be entries
! for the EIGRP multicast as well as for the office’s own unicast address.
access-list 111 permit igrp any any
!
! These are the ICMP types actually used...
! administratively-prohibited is useful when you are trying to figure out why
! you cannot reach something you think you should be able to reach.
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 administratively-prohibited
!
! This allows network admins at headquarters to ping hosts at the field office:
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 echo
!
! This allows the field office to do outgoing pings
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 echo-reply
!
! Path MTU discovery requires too-big messages
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 packet-too-big
!
! Outgoing traceroute requires time-exceeded messages to come back
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 time-exceeded
!
! Incoming traceroute
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 traceroute
!
! Permits all unreachables because if you are trying to debug
! things from the remote office, you want to see them. If nobody ever did
! any debugging from the network, it would be more appropriate to permit only
! port unreachables or no unreachables at all.
access-list 111 permit icmp any 172.19.139.0 0.0.0.7 unreachable
!
!
! These next two entries permit users on most ExampleCorp networks to Telnet to
! a host in the field office. This is for remote administration by the network admins.
access-list 111 permit tcp 172.18.0.0 0.1.255.255 host 172.19.139.1 eq telnet
access-list 111 permit tcp 192.168.1.0 0.0.0.255 host 172.19.139.1 eq telnet
!
! Final deny for explicitness
access-list 111 deny ip any any
!
no cdp run
snmp-server community <elided> RO
!
line con 0
exec-timeout 0 0
password <elided>
login local
line vty 0
exec-timeout 0 0
password <elided>
```

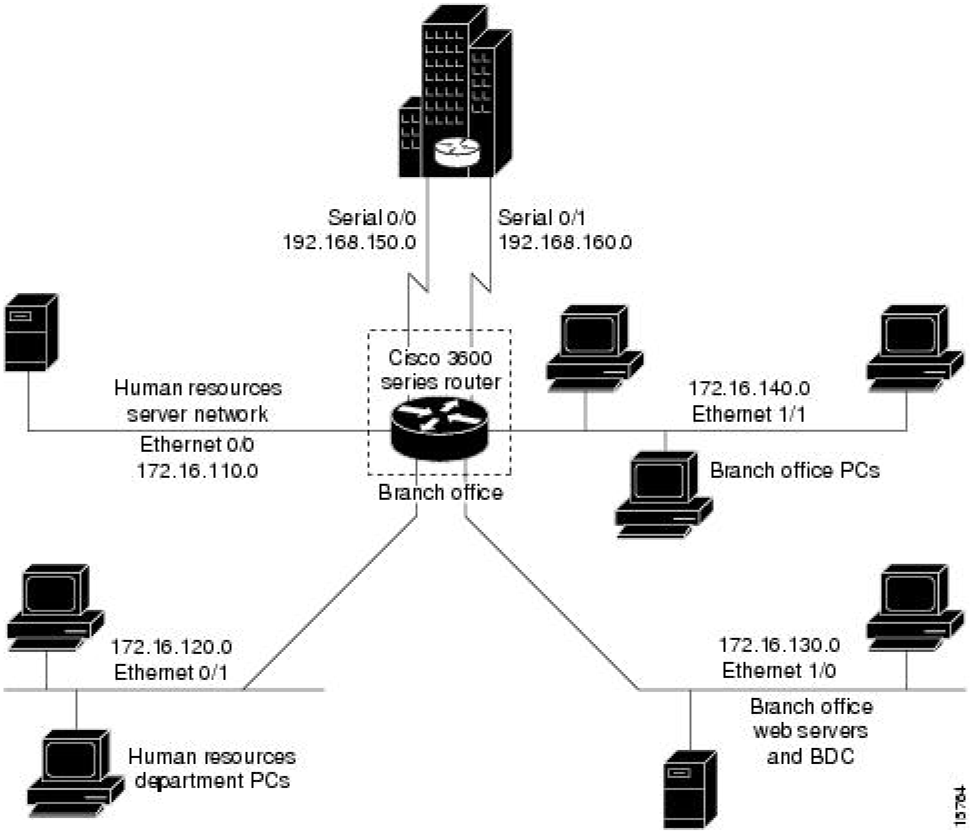
```
login local
length 35
line vty 1
exec-timeout 0 0
password 7 <elided>
login local
line vty 2
exec-timeout 0 0
password 7 <elided>
login local
line vty 3
exec-timeout 0 0
password 7 <elided>
login local
line vty 4
exec-timeout 0 0
password 7 <elided>
login local
!
scheduler interval 500
end
```

Multiple-Interface Branch Office Configuration Example

In this configuration example, a single Cisco 3600 series firewall router is positioned at a branch office. It has four internal networks and two WAN connections to the corporate headquarters. CBAC is configured on the firewall to protect two of the internal networks from potential network threats coming from the WAN side and from less secure internal networks. Anti-spoofing protection is added at each interface with client systems. The figure below illustrates this configuration.

Note This example shows a moderately high level of trust by the administrators toward the expected users. Additional protection could be added to this configuration for a situation in a lower level of trust. That configuration would include ICMP filtering statements, significantly more protocol and address control through the use of more restrictive access control lists, and anti-spoofing applied everywhere. This configuration does not contain those additional restrictions because that would detract from the CBAC example.

Figure 6: Sample Cisco IOS Firewall Application Environment



The branch office has this sample network configuration:

- Ethernet interface 0/0 supports the Human Resources department servers. This network includes an email (SMTP and POP3) host and a Windows NT server. The Windows NT server is the Primary Domain Controller (PDC) for the Human Resources domain and has a trust relationship with the rest of the company; however, it contains applications and databases that must not be accessed by the rest of the company or the other groups in the branch office. The devices on this LAN are accessible only by users in the Human Resources department on Ethernet interface 0/1. The Mail server must be able to send and receive email (through SMTP sessions) with all other devices. The Windows 95 machines can use this machine as their email server (for sending email through SMTP sessions) and as a repository for accumulating email that they can then download through POP3 sessions. No one else in the company is allowed to form POP3 sessions to any machine on this LAN.

- Ethernet interface 0/1 supports the Windows 95 computers in the Human Resources department. These users must have access to the Human Resources mail servers located on Ethernet interface 0/0 as well as access to the rest of the company. Access to the Windows NT server resources are controlled through the Windows NT permissions assigned to each user in the Windows NT domain.
- Ethernet interface 1/0 supports the branch office web servers, which can be accessed by everyone in the company. These servers use TCP ports 80 (HTTP) and 443 (SHTTP) for inbound Web access. This network also includes a backup domain controller (BDC) for the overall domain that is also used as file, print, and service server.

Ethernet interface 1/1 supports all users who are not in the Human Resources department. These users have no access to the Human Resources department servers, but they can access the other network interfaces and the serial interfaces for WAN connectivity. Serial interface 0/0 and 0/1 connect to the WAN with T1 links (links to corporate headquarters). In this sample configuration, the Domain Name System (DNS) servers are located somewhere within the rest of the company.

Additionally, network management (SNMP) and Telnet sessions are limited to the management network (192.168.55.0), which is located somewhere within the rest of the company across the serial interface.

```
! -------------------------------------------------------------------
! This first section contains some configuration that is not required
! for CBAC, but illustrates good security practices.
! -------------------------------------------------------------------
! Add this line to get timestamps on the syslog messages.
service timestamps log datetime localtime show-timezone
!
hostname Router1
!
boot system flash c3600-fw3600-l
!
! Configure AAA user authentication.
aaa new-model
aaa authentication login lista group tacacs+ enable
!
enable secret 5 <elided>
ip subnet-zero
!
! Disable source routing to help prevent spoofing.
no ip source-route
!
! Set up the domain name and server IP addresses.
ip domain-name example.com
ip name-server 192.168.55.132
ip name-server 192.168.27.32
!
! The audit-trail command enables the delivery of specific CBAC messages
! through the syslog notification process.
ip inspect audit-trail
!
! Establish the time-out values for DNS queries. When this idle-timer expires,
! the dynamic ACL entries that were created to permit the reply to a DNS request
! will be removed and any subsequent packets will be denied.
ip inspect dns-timeout 10
!
! ----------------------------------------------------------------------------------
! The next section includes configuration statements required specifically for CBAC.
! ----------------------------------------------------------------------------------
! Define the CBAC inspection rule “inspect1”, allowing the specified protocols to be
! inspected. The first rule enables SMTP specific inspection. SMTP inspection causes
! the exchange of the SMTP session to be inspected for illegal commands. Any packets
! with illegal commands are dropped, and the SMTP session will hang and eventually
! time out.
ip inspect name inspect1 smtp timeout 30
!
! In the next two lines of inspect1, define the maximum time that each of the UDP and
! TCP sessions are allowed to continue without any traffic passing
! through the router. When these timeouts are reached, the dynamic ACLs that
! are inserted to permit the returning traffic are removed and subsequent packets
! (possibly even valid ones) will not be permitted.
ip inspect name inspect1 udp timeout 30
ip inspect name inspect1 tcp timeout 30
!
! Define the CBAC inspection rule “inspect2”, allowing the specified protocols to be
! inspected. These rules are similar to those used in the inspection rule “inspect1,”
! except that on the interfaces where this rule is applied, SMTP sessions are not
! expected to go through; therefore, the SMTP rule element is not applied here.
ip inspect name inspect2 udp timeout 30
ip inspect name inspect2 tcp timeout 30
!
! -------------------------------------------------------------------
! The next section shows the Ethernet interface configuration statements for each
! interface, including access lists and inspections rules.
! -------------------------------------------------------------------
! Apply the “inspect1” inspection rule to sessions that are initiated in the outbound
! direction (toward the LAN) at Ethernet interface 0/0. All packets in these sessions
! will be inspected by CBAC. Provided that network traffic passes the Access Control
! List (ACL) restrictions, traffic is then inspected by CBAC for access through the
! Cisco Secure Integrated Software. Traffic blocked by the access list is not inspected
! by CBAC. Access list 110 is applied to outbound traffic on this interface.
interface Ethernet0/0
description HR_Server Ethernet
ip address 172.16.110.1 255.255.255.0
ip access-group 110 out
no ip directed-broadcast
no ip proxy-arp
ip inspect inspect1 out
no cdp enable
!
! Apply access list 120 to inbound traffic on Ethernet interface 0/1.
! Applying access list 120 to inbound traffic provides anti-spoofing on this interface
! by dropping traffic with a source address matching the IP address on a network other
! than Ethernet 0/1. The IP helper address lists the IP address of the DHCP server on
! Ethernet interface 1/0.
interface Ethernet0/1
description HR_client Ethernet
ip address 172.16.120.1 255.255.255.0
ip access-group 120 in
ip helper-address 172.16.130.66
no ip directed-broadcast
no ip proxy-arp
no cdp enable
!
! Apply the “inspect2” inspection rule to sessions that are initiated in the outbound
! direction (toward the LAN) at Ethernet interface 1/0. Provided that network traffic
! passes the Access Control List (ACL) restrictions, traffic is then inspected by CBAC
! through the Cisco Secure Integrated Software. Traffic blocked by the access list is
! not inspected
! by CBAC. Access list 130 is applied to outbound traffic on this interface.
interface Ethernet1/0
description Web_server Ethernet
ip address 172.16.130.1 255.255.255.0
ip access-group 130 out
no ip directed-broadcast
no ip proxy-arp
ip inspect inspect2 out
no cdp enable
!
! Apply access list 140 to inbound traffic at Ethernet interface 1/1. This
! provides anti-spoofing on the interface by dropping traffic with a source address
! matching the IP address of a network other than Ethernet 1/1. The IP helper address
! lists the IP address of the DHCP server on Ethernet interface 1/0.
interface Ethernet1/1
description Everyone_else Ethernet
ip address 172.16.140.1 255.255.255.0
ip access-group 140 in
ip helper-address 172.16.130.66
no ip directed-broadcast
no ip proxy-arp
no cdp enable
!
! --------------------------------------------------------------------------
! The next section configures the serial interfaces, including access lists.
! --------------------------------------------------------------------------
! Apply access list 150 to Serial interfaces 0/0. This provides anti-spoofing on the
! serial interface by dropping traffic with a source address matching the IP address
! of a host on Ethernet interface 0/0, 0/1, 1/0, or 1/1.
interface Serial0/0
description T1 to HQ
ip address 192.168.150.1 255.255.255.0
ip access-group 150 in
bandwidth 1544
!
interface Serial1/1
description T1 to HQ
ip address 192.168.160.1 255.255.255.0
ip access-group 150 in
bandwidth 1544
!
! ------------------------------
! Configure routing information.
! ------------------------------
router igrp 109
network 172.16.0.0
network 192.168.150.0
network 192.168.160.0
!
! Define protocol forwarding on the firewall. When you turn on a related command,
! ip helper-address, you forward every IP broadcast in the ip forward protocol
! command list, including several which are on by default: TFTP (port 69),
! DNS (port 53), Time service (port 37), NetBIOS Name Server (port 137),
! NetBIOS Datagram Server (port 138), BOOTP client and server datagrams
! (ports 67 and 68), and TACACS service (port 49). One common
! application that requires helper addresses is Dynamic Host Configuration
! Protocol (DHCP). DHCP information is carried inside of BOOTP packets. The
! “no ip forward protocol” statements turn off forwarding for the specified protocols.
no ip forward-protocol udp netbios-ns
no ip forward-protocol udp netbios-dgm
no ip forward-protocol udp tacacs
no ip forward-protocol udp tftp
ip forward-protocol udp bootpc
!
! Add this line to establish where router SYSLOG messages are sent. This includes the
! CBAC messages.
logging 192.168.55.131
!
! ---------------------------------------------------------------
! Define the configuration of each access list.
! ---------------------------------------------------------------
! Defines Telnet controls in access list 12.
access-list 12 permit 192.168.55.0 0.0.0.255
!
! Defines SNMP controls in access list 13.
access-list 13 permit 192.168.55.12
access-list 13 permit 192.168.55.19
!
! Access list 110 permits TCP and UDP protocol traffic for specific ports and with a
! source address on Ethernet interface 0/1. The access list denies IP protocol traffic
! with any other source and destination address. The access list permits ICMP access
! for any source and destination address. Access list 110 is deliberately set up to
! deny unknown IP protocols because no such unknown protocols will be in legitimate
! use. Access list 110 is applied to outbound traffic at Ethernet interface 0/0. In ACL
! 110, network traffic is being allowed access to the ports on any server on the HR
! server network. In less trusted environments, this can be a security problem;
! however, you can limit access more severely by specifying specific destination
! addresses in the ACL statements.
access-list 110 permit tcp 172.16.120.0 0.0.0.255 any eq smtp
access-list 110 permit tcp 172.16.120.0 0.0.0.255 any eq pop3
access-list 110 permit tcp 172.16.120.0 0.0.0.255 any eq 110
access-list 110 permit udp any any eq 137
access-list 110 permit udp any any eq 138
access-list 110 permit udp any any eq 139
```

```
access-list 110 permit icmp any any
access-list 110 deny ip any any!
!
! Access-list 120 permits TCP, UDP, and ICMP protocol traffic with a source address
! on Ethernet interface 0/1, but denies all other IP protocol traffic. Access list
! 120 is applied to inbound traffic on Ethernet interface 0/1.
access-list 120 permit tcp 172.16.120.0 0.0.0.255 any
access-list 120 permit udp 172.16.120.0 0.0.0.255 any
access-list 120 permit icmp 172.16.120.0 0.0.0.255 any
access-list 120 deny ip any any
!
! Access list 130 permits TCP, UDP, and ICMP protocol traffic for specific ports and
! with any source and destination address. It opens access to the web server and to
! all NBT services to the rest of the company, which can be controlled through the
! trust relations on the Windows NT servers. The bootpc entry permits access to the
! DHCP server. Access list 130 denies all other IP protocol traffic. Access list 130 is
! applied to outbound traffic at Ethernet interface 1/0.
access-list 130 permit tcp any any eq www
access-list 130 permit tcp any any eq 443
access-list 130 permit tcp any any eq 110
access-list 130 permit udp any any eq 137
access-list 130 permit udp any any eq 138
access-list 130 permit udp any any eq 139
access-list 130 permit udp any any eq bootpc
access-list 130 permit icmp any any
access-list 130 deny ip any any
!
! Access list 140 permits TCP, UDP, and ICMP protocol traffic with a source address on
! Ethernet interface 1/1, and it denies all other IP protocol traffic. Access list 140
! is applied to inbound traffic at Ethernet interface 1/1.
access-list 140 permit tcp 172.16.140.0 0.0.0.255 any
access-list 140 permit udp 172.16.140.0 0.0.0.255 any
access-list 140 permit icmp 172.16.140.0 0.0.0.255 any
access-list 140 deny ip any any
!
! Access list 150 denies IP protocol traffic with a source address on Ethernet
! interfaces 0/0, 0/1, 1/0, and 1/1, and it permits IP protocol traffic with any other
! source and destination address. Access list 150 is applied to inbound traffic
! on each of the serial interfaces.
access-list 150 deny ip 172.16.110.0 0.0.0.255 any
access-list 150 deny ip 172.16.120.0 0.0.0.255 any
access-list 150 deny ip 172.16.130.0 0.0.0.255 any
access-list 150 deny ip 172.16.140.0 0.0.0.255 any
access-list 150 permit ip any any
!
! Disable Cisco Discovery Protocol.
no cdp run
!
snmp-server community <elided> ro 13
tacacs-server host 192.168.55.2
tacacs-server key <elided>
!
! ---------------------------------------------------------------------------------
! Configures the router console port and the virtual terminal line interfaces,
! including AAA authentication at login. Authentication is required for users defined
! in “lista.” Access-class 12 is applied on each line, restricting Telnet access to
! connections with a source address on the network management network.
! ---------------------------------------------------------------------------------
line console 0
exec-timeout 3 00
login authentication lista
line aux 0
exec-timeout 3 00
login authentication lista
line vty 0
exec-timeout 1 30
login authentication lista
access-class 12 in
line vty 1
exec-timeout 1 30
login authentication lista
access-class 12 in
line vty 2
```

```
exec-timeout 1 30
login authentication lista
access-class 12 in
line vty 3
exec-timeout 1 30
login authentication lista
access-class 12 in
line vty 4
exec-timeout 1 30
login authentication lista
access-class 12 in
!
end
```

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

CHAPTER 2

IPv6 IOS Firewall

The IPv6 IOS Firewall feature supports the inspection of IPv6 packets. With IPv6 support, the Cisco firewall inspects both IPv4 and IPv6 packets on devices with dual stacks. Devices that support IPv4 and IPv6 packet inspection are called dual-stack devices. This feature also provides MIB support for FTP, Internet Control Message Protocol Version 6 (ICMPv6), TCP, and UDP sessions.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for IPv6 IOS Firewall

- Cisco Intrusion Prevention System (previously known as Cisco Intrusion Detection System) is not supported for IPv6.
- Cisco Context-Based Access Control (CBAC) firewalls do not inspect IPv6 router-generated traffic.

Information About IPv6 IOS Firewall

Cisco IOS Firewall for IPv6

The Cisco IOS firewall does advanced traffic filtering as an integral part of a network. The IPv6 IOS Firewall feature enables you to implement the IOS firewall in IPv6 networks. The IPv6 IOS firewall coexists with the IOS firewall for IPv4 networks and is supported on all dual-stack devices.

The IPv6 firewall provides the following features:

- Fragmented packet inspection—A fragment header is used to trigger fragment processing. Virtual Fragment Reassembly (VFR) examines out-of-sequence fragments and switches packets into the correct order, examines the number of fragments from a single IP that have a unique ID (denial of service [DoS] attacks), and performs virtual reassembly to move packets to upper-layer protocols.
- IPv6 DoS attack mitigation—Mitigation mechanisms to prevent DoS attacks including synchronized (SYN) half-open connections.
- Tunneled packet inspection—Tunneled IPv6 packets terminated at a device configure with firewall can be inspected by the IPv6 IOS firewall.
- Stateful packet inspection—Stateful packet inspection of FTP, Internet Control Message Protocol version 6 (ICMPv6), TCP, and UDP sessions. Also provides the stateful inspection of packets originating from an IPv4 network and terminating in an IPv6 environment. This feature uses IPv4-to-IPv6 translation services.
- Interpretation or recognition of most IPv6 extension header information—IPv6 extension header information including routing header, hop-by-hop options header, and fragment header is interpreted or recognized.
- Port-to-Application Mapping (PAM)—The IPv6 IOS firewall supports PAM.

PAM in Cisco IOS IPv6 Firewall

Port-to-Application Mapping (PAM) allows you to customize TCP or UDP port numbers for network services or applications. PAM uses this information to support network environments that run services using ports that are different from the registered or well-known ports associated with an application.

Using the port information, PAM establishes a table of default Port-to-Application Mapping information at the firewall. The information in the PAM table enables Context-Based Access Control (CBAC) supported services to run on nonstandard ports. CBAC is limited to inspecting traffic using only the well-known or registered ports associated with an application, whereas PAM allows network administrators to customize network access control for specific applications and services.

PAM also supports host- or subnet-specific port mapping, which allows you to apply PAM to a single host or a subnet using standard access control lists (ACLs). Host- or subnet-specific port mapping is done using standard ACLs.

Alerts, Audit Trails, and System Logging in Firewalls

The Cisco IOS firewall generates real-time alerts and audit trails based on events tracked by the firewall. Enhanced audit trail features use system logging to track all network transactions such as time stamps, source host, destination host, ports used, and to record the total number of transmitted bytes for advanced, session-based reporting. Real-time alerts send system logging error messages to central management consoles when the system detects any suspicious activity. Using Cisco IOS firewall inspection rules, you can configure alerts and audit trail information on a per-application protocol basis. For example, to generate audit trail information for TCP traffic, specify the generation of this information in the Cisco IOS firewall rule that defines TCP inspection.

The Cisco IOS firewall provides audit trail messages to record details about inspected sessions. Audit trail information is configurable on a per-application basis using Context-Based Access Control (CBAC) inspection rules. To determine the protocol that is inspected, use the port number associated with the responder. The port number appears immediately after the IP address.

IPv6 Packet Inspection

The following header fields are used for IPv6 inspection: traffic class, flow label, payload length, next header, hop limit, and source or destination IP address. For further information on and descriptions of the IPv6 header fields, see RFC 2474.

Tunneling Support

IPv6 packets tunneled in IPv4 are not inspected. If a tunnel terminates on a device, and the IPv6 traffic exiting the tunnel is nonterminating, then that traffic is inspected.

Virtual Fragment Reassembly

When Virtual Fragment Resassembly (VFR) is enabled, the VFR processing begins after access control list (ACL) input lists are checked against incoming packets. Incoming packets are tagged with the appropriate VFR information.

How to Configure IPv6 IOS Firewall

Configuring the Cisco IOS Firewall for IPv6

This configuration scenario uses both packet inspection and access control lists (ACLs).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **ipv6 inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]
5. **interface** *type number*
6. **ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}
7. **ipv6 enable**
8. **ipv6 traffic-filter** *access-list-name* {**in** | **out**}
9. **ipv6 inspect** *inspection-name* {**in** | **out**}
10. **exit**
11. **ipv6 access-list** *access-list-name*
12. Do one of the following:
    - **permit** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
    - **deny** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables IPv6 unicast routing.

	Command or Action	Purpose
Step 4	**ipv6 inspect name** *inspection-name protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*] **Example:** `Device(config)# ipv6 inspect name ipv6-test icmp timeout 60`	Defines a set of IPv6 inspection rules for the firewall.
Step 5	**interface** *type number* **Example:** `Device(config)# interface FastEthernet 0/0`	Specifies the interface on which the inspection will occur.

Step 6	**ipv6 address** {*ipv6-address/prefix-length* | *prefix-name sub-bits/prefix-length*}
	Example:
	`Device(config-if)# ipv6 address 2001:DB8::1/64 eui-64`
		Specifies an IPv6 address for the inspection interface.

Step 7	**ipv6 enable** **Example:** `Device(config-if)# ipv6 enable`	Enables IPv6 routing. **Note** This step is optional if the IPv6 address is specified in step 6.

Step 8	**ipv6 traffic-filter** *access-list-name* {**in** | **out**}
	Example:
	`Device(config-if)# ipv6 traffic-filter outbound out`
		Applies the specified IPv6 access list to the interface specified in Step 5.

Step 9	**ipv6 inspect** *inspection-name* {**in** | **out**}
	Example:
	`Device(config-if)# ipv6 inspect ipv6-test in`
		Applies the set of inspection rules.

Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list outbound`	Defines an IPv6 ACL and enters IPv6 access list configuration mode.
Step 12	Do one of the following: • **permit** *protocol* {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**]	Specifies permit or deny conditions for an IPv6 ACL.

	Command or Action	Purpose

	[**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
	• **deny** *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
	Example:
	`Device(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 any reflect reflectout`
	Example:
	`Device(config-ipv6-acl)# deny tcp 2001:DB8:0:ABCD::1/64 any`	

Step 13	**end**
	Example:
	`Device(config-ipv6-acl)# end`
		Exits IPv6 access list configuration mode and returns to privileged EXEC mode.

Configuring PAM for IPv6

Creating an IPv6 Access Class Filter for PAM

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
   - **permit** *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix* /*prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
   - **deny** *protocol source-ipv6-prefix*/*prefix-length* | **any** |**host** *source-ipv6-address* | **auth**} [*operator port-number*]] *destination-ipv6-prefix/prefix-length* **any host** *destination-ipv6-address* | **auth**} [*operator port-number*]] **dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list outbound`	Defines an IPv6 access control list (ACL) and enters IPv6 access list configuration mode.

	Command or Action	Purpose

Step 4	Do one of the following:
	• **permit** *protocol* {*source-ipv6-prefix*/*prefix-length* | **any** | **host** *source-ipv6-address* | **auth**} [*operator* [*port-number*]] {*destination-ipv6-prefix* /*prefix-length* | **any** | **host** *destination-ipv6-address* | **auth**} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**reflect** *name* [**timeout** *value*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
	• **deny** *protocol source-ipv6-prefix/prefix-length* | **any** |**host** *source-ipv6-address* | **auth**} [*operator port-number*]] *destination-ipv6-prefix/prefix-length* **any host** *destination-ipv6-address* | **auth**} [*operator port-number*]] **dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]
	Example:
	`Device(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 any reflect reflectout`
	Example:
	`Device(config-ipv6-acl)# deny tcp 2001:DB8:0:ABCD::1/64 any`
		Specifies permit or deny conditions for an IPv6 ACL.

Step 5	**end**
	Example:
	`Device(config-ipv6-acl)# end`
		Exits IPv6 access list configuration mode and returns to privileged EXEC mode.

Applying the IPv6 Access Class Filter to PAM

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 port-map** *application-name* **port** *port-num* [**list** *acl-name*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 port-map** *application-name* **port** *port-num* [**list** *acl-name*] **Example:** `Device(config)# ipv6 port-map ftp port 8090 list PAMACL`	Establishes PAM for the system.

Step 4	**end**
	Example:
	`Device(config)# end`
		Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for IPv6 IOS Firewall

Example: Configuring Cisco IOS Firewall for IPv6

The following configuration example uses inbound and outbound filters for inspection and makes use of access lists to manage traffic. The inspect mechanism is the method of permitting return traffic based upon a packet being valid for an existing session for which the state is being maintained:

```
configure terminal
 ipv6 unicast-routing
  ipv6 inspect name ipv6_test icmp timeout 60
  ipv6 inspect name ipv6_test tcp timeout 60
  ipv6 inspect name ipv6_test udp timeout 60
!
 interface FastEthernet 0/0
  ipv6 address 2001:DB8::1/32 eui-64
  ipv6 enable
  ipv6 traffic-filter INBOUND out
  ipv6 inspect ipv6-test in
!
 interface FastEthernet 0/1
  ipv6 address 2001:DB8:0000:FFFF:FFFF:FFFF:FFFF: FFFF /64 eui-64
  ipv6 enable
  ipv6 traffic-filter OUTBOUND in
!
! This is used for 3745b connection to tftpboot server
```

```
 interface FastEthernet 4/0
  ip address 192.168.17.33 255.255.255.0
  duplex auto
  speed 100
!
 ip default-gateway 192.168.17.8
! end of tftpboot server config

! Access-lists to deny everything except for Neighbor Discovery ICMP messages
 ipv6 access-list INBOUND
  permit icmp any any nd-na
  permit icmp any any nd-ns
  deny ipv6 any any log
!
 ipv6 access-list OUTBOUND
  permit icmp any any nd-na
  permit icmp any any nd-ns
  deny ipv6 any any log
!
```

Additional References for IPv6 IOS Firewall

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z
IPv6 commands	Cisco IOS IPv6 Command Reference
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS IPv6 features	Cisco IOS IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	*IPv6 RFCs*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

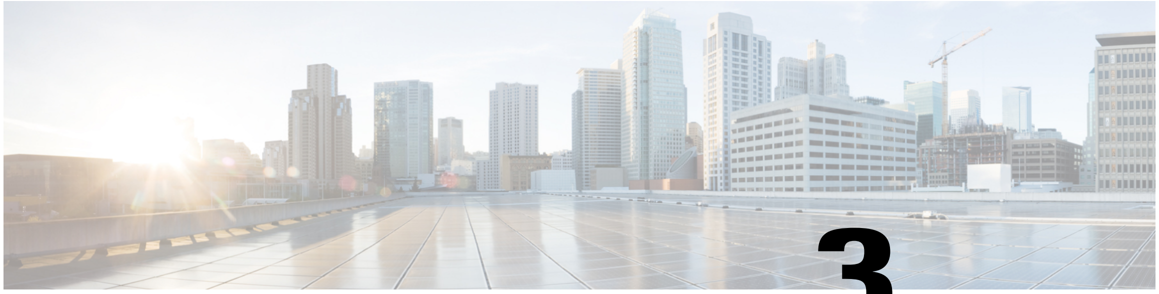
Feature Information for IPv6 IOS Firewall

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for IPv6 IOS Firewall

Feature Name	Releases	Feature Information
IPv6 IOS Firewall	12.3(7)T	The IPv6 IOS Firewall feature provides advanced traffic filtering functionality as an integral part of a network’s firewall.
IPv6 Services—IPv6 IOS Firewall FTP Application Support	12.3(11)T	IPv6 supports FTP application support.



CHAPTER 3

Cisco IOS Firewall Stateful Failover

Stateful failover for the Cisco IOS firewall enables a router to continue processing and forwarding firewall session packets after a planned or unplanned outage occurs. You employ a backup (secondary) router that automatically takes over the tasks of the active (primary) router if the active router loses connectivity for any reason. This process is transparent and does not require adjustment or reconfiguration of any remote peer.

Stateful failover for the Cisco IOS firewall is designed to work in conjunction with stateful switchover (SSO) and Hot Standby Routing Protocol (HSRP). HSRP provides network redundancy for IP networks, ensuring that user traffic immediately and transparently recovers from failures in network edge devices or access circuits. That is, HSRP monitors both the inside and outside interfaces so that if either interface goes down, the whole router is deemed to be down and ownership of firewall sessions is passed to the standby router (which transitions to the HSRP active state). SSO allows the active and standby routers to share firewall session state information so that each router has enough information to become the active router at any time. To configure stateful failover for the Cisco IOS firewall, a network administrator should enable HSRP, assign a virtual IP address, and enable the SSO protocol.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Stateful Failover

Complete, Duplicate Cisco IOS Firewall Configuration on the Active and Standby Devices

This document assumes that you have a complete Cisco IOS firewall configuration.

The Cisco IOS firewall configuration that is set up on the active device must be duplicated on the standby device. That is, firewall protocols inspected, the interface ACL's, the global firewall settings and the interface firewall configuration.

Note None of the configuration information between the active and standby device is automatically transferred; the user is responsible for ensuring that the Cisco IOS firewall configurations match on both devices. If the Cisco IOS firewall configurations on both devices do not match, failover from the active device to the standby device will not be successful.

Device Requirements

- The active and standby Cisco IOS routers must be running the same Cisco IOS software, Release 12.4(6)T or later.
- Stateful failover for the Cisco IOS firewall requires that your network contains two identical routers that are available to be either the primary or secondary device. Both routers should be the same type of device, have the same CPU and memory.
- This feature is currently supported only on a limited number of platforms. To check the latest platform support, go to Cisco Feature Navigator at http://www.cisco.com/go/fn .

Restrictions for Stateful Failover

When configuring redundancy for a Cisco IOS firewall, the following restrictions exist:

- Both the active and standby devices must run the identical version of the Cisco IOS software, and both the active and standby devices must be connected via hub or switch.
- HSRP requires the inside interface to be connected via LANs.
- Load balancing is not supported; that is, no more than one device in a redundancy group can be active at any given time.
- Any restrictions that exist for intradevice SSO will also exist for the firewall High Availability (HA). The behavior of intra-device active where the Active device re-boots when the SSO state changes from Active to anything will be the same with firewall HA.
- No support for configuration synchronization and In-Service Software Upgrade (ISSU) which are not yet available for intra-box failover in Cisco IOS T releases.
- Stateful failover of the Cisco IOS firewall is not supported with Zone-Based Policy firewall configuration.

- This phase of the feature will not provide support for asymmetric routing and it is the responsibility of the user to configure the network to avoid this.
- The stateful failover feature does not synchronize any statistics or mib firewall information between the active and standby devices.
- The stateful failover feature does not support rate-limiting of firewall sessions on the standby router for the failed over sessions.
- Currently only Layer 4 TCP and UDP protocol failover is supported. Therefore, all TCP only sessions, UDP only sessions, and single channel granular protocols sessions for which L7 inspection is not supported are failed over.
- Layer 4 ICMP session will not be failed over to the standby
- Any session configured for Layer 7 inspection will NOT be failed over.
- CiscoIntrusion Prevention Services (IPS)/Intrusion Detection Services (IDS) feature will not be made HA aware in this release.

Information About Stateful Failover

Supported Deployment Scenarios Stateful Failover for the Cisco IOS Firewall

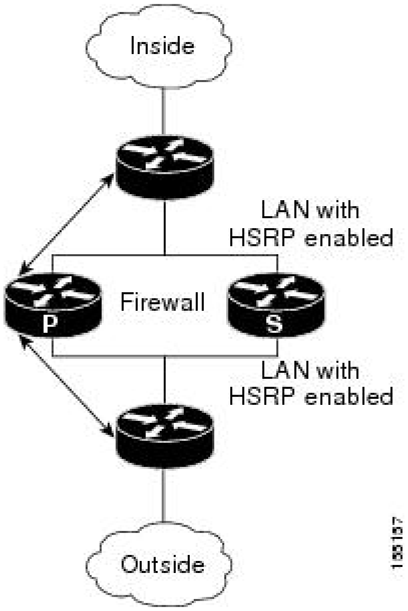
It is recommended that you implement stateful failover in one of the following recommended deployment scenarios:

- Dual LAN Interface
- LAN WAN Interface

In a dual LAN interface scenario, the active and standby routers running the firewall are connected to each other via LAN interface on both the inside and the outside (see the figure below). HSRP is configured on both the inside and outside interface. The next hop routers in this scenario talk to the HA pair via the virtual IP address. In this scenario there are two virtual IP address, one on the inside and the other on the outside. Virtual IP addresses cannot be advertised using routing protocols. You need to create static routes on the next hops to get to the virtual IP address.

You need to configue HSRP tracking in order to track multiple pairs of interfaces. If you run HSRP on only one pair of interfaces, or run on both without mutual tracking of the pairs, each pair functions independently of each other and are unaware of each other's state changes. For example, if HSRP is run on only the two outside interfaces (as shown in the figure below), this could cause HSRP to failover on the outside interfaces, whilst the inside interfaces are unchanged. This causes the black holing of traffic, which continues to be directed to the primary from the inside. This introduces the possibility of problems arising from one interface on a primary router failing and triggering a move to the secondary, while the other interface on the ex-primary remains active. Mutual tracking means that if the outside interface does fail, the inside interface on the same router will also be deemed down allowing for complete router failover to the secondary.

Figure 7: Dual Interface Network Topology



In a LAN WAN scenario, the inside interface of the Active Standby pair running the firewall are connected via LAN interface on the inside and WAN interface on the outside (see the figure below). HSRP is configured on the inside interface. The inside network communicates with the HA pair using the inside virtual IP address.

HSRP tracking should be configured on the inside LAN interfaces to track the state of the outside WAN interface. If the outside WAN interface goes down on the active the LAN interface that is tracking it reduces the HSRP priority and initiates a failover to the standby. Traffic from the outside flowing into the HSRP pair should now be directed to the new active device.

In the scenario where the LAN interfaces track the WAN interfaces, the failover to the standby happens immediately. However, for traffic to start flowing on the new active router, routing convergence needs to happen. The net failover time is dictated by the routing protocol.

In this scenario the traffic flows from the inside to the outside through the Active due to the HSRP configuration on the inside LAN interfaces. The traffic from the outside to the inside should also flow through the active device. The configuration of the network so that the traffic always flows through the active is beyond the scope of this document. In this scenario, the network administrator is responsible to ensure that the traffic always flows through the active device.

Stateful Failover Architecture

Firewall stateful failover is a client of Cisco IOS SSO. SSO is a method of providing redundancy and synchronization for Cisco IOS applications and features.

State Synchronization

The synchronization manager will be responsible for checking firewall to determine the state of the active device, which must be check pointed to the redundant peers and update that state on the firewall on standby devices.

Periodic updates are sent from the active to the standby for all HA sessions. This information enables the standby to take over the sessions and process the sessions if there is a failover.

The stateful failover feature supports deterministic updates. This means that the updates for a session get sent every N seconds, where N is configurable. Default value for N is 10 sec.

Bulk Synchronization

Bulk synchronization happens at boot time or when you use the **clear ip inspect ha sessions all**command on the standby device. If the standby device is configured after the active device already has sessions, then only new ha sessions established on the active device are synchronized to the standby device through dynamic synchronization. If you want all the current sessions synchronized from the active to the standby, you must specifically issue the **clear ip inspect ha sessions all**command on the standby device. A single request message is sent from the standby device to the active device which result in add_session messages from active to standby for all sessions open on the active at that time.

How to Configure Stateful Failover for Cisco IOS Firewalls

Enabling HSRP IP Redundancy and a Virtual IP Address

HSRP provides two services--IP redundancy and a Virtual IP (VIP) address. Each HSRP group may provide either or both of these services. Cisco IOS firewall stateful failover uses the IP redundancy services from only one HSRP standby group. It can use the VIP address from one or more HSRP groups. Use the following task to configure HSRP on the outside and inside interfaces of the router.

Note Perform this task on both routers (active and standby) and on both interfaces of each router.

Before You Begin

If a switch connects the active and standby routers, you must perform one of the following steps to ensure that the correct settings are configured on that switch:

- Enable the **spanning-tree portfast** command on every switch port that connects to a HSRP-enabled router interface.
- Disable the Spanning Tree Protocol (STP) on the switch only if your switch does not connect to other switches. Disabling spanning tree in a multi-switch environment may cause network instability.
- Enable the **standby delay minimum** [*min-delay*] **reload** [*reload-delay*] command if you do not have access to the switch. The *reload-delay* argument should be set to a value of at least 120 seconds. This command must be applied to all HSRP interfaces on both routers.

For more information on HSRP instability, see the document Avoiding HSRP Instability in a Switching Environment with Various Router Platforms .

Note You must perform at least one of these steps for correct HSRP operation.

Note

- Both the inside (private) interface and the outside (public) interface must belong to separate HSRP groups, but the HSRP group number can be the same.
- The state of the inside interface and the outside interface must be the same--both interfaces must be in the active state or standby state; otherwise, the packets will not have a route out of the private network.
- Standby priorities should be equal on both active and standby routers. If the priorities are not equal, the higher priority router will unnecessarily take over as the active router, negatively affecting uptime.
- The IP addresses on the HSRP-tracked interfaces of the standby and active routers should both be either lower or higher on one router than the other. In the case of equal priorities (an HA requirement), HSRP will assign the active state on the basis of the IP address. If an addressing scheme exists so that the public IP address of Router A is lower than the public IP address of Router B, but the opposite is true for their private interfaces, an active/standby-standby/active split condition could exist which will break connectivity.
- Interface ACL should allow HSRP traffic to flow through.
- Each time an active device relinquishes control to become the standby device, the active device will reload. This functionality ensures that the state of the new standby device synchronizes correctly with the new active device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. interface *type number*
4. standby standby-group-number name standby-group-name
5. standby standby-group-number ip ip-address
6. standby standby-group-number track interface-name
7. **standby** [*group-number*] **preempt**
8. **standby** [*group-number*] **timers** [**msec**] *hellotime* [**msec**] *holdtime*
9. **standby delay minimum** [*min-delay*] **reload** [*reload-delay*]
10. Repeat.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	interface *type number* **Example:** `Router(config)# interface Ethernet 0/0`	Configures an interface type for the router and enters interface configuration mode.
Step 4	standby standby-group-number name standby-group-name **Example:** `Router(config-if)# standby 1 name HA-out`	Assigns a user-defined group name to the HSRP redundancy group. **Note** The *standby-group-number* argument should be the same for both routers that are on directly connected interfaces. However, the *standby-group-name* argument should be different between two (or more) groups on the same router. The *standby-group-number* argument can be the same on the other pair of interfaces as well.
Step 5	standby standby-group-number ip ip-address **Example:** `Router(config-if)# standby 1 ip 209.165.201.1`	Assigns an IP address that is to be “shared” among the members of the HSRP group and owned by the primary IP address. **Note** The virtual IP address must be configured identically on both routers (active and standby) that are on directly connected interfaces.
Step 6	standby standby-group-number track interface-name **Example:** `Router(config-if)# standby 1 track Ethernet1/0`	Configures HSRP to monitor the second interface so that if either of the two interfaces goes down, HSRP causes failover to the standby device. **Note** Although this command is not required, it is recommended for dual interface configurations.
Step 7	**standby** [*group-number*] **preempt** **Example:** `Router(config-if)# standby 1 preempt`	Enables the active device to relinquish control because of an interface tracking event.
Step 8	**standby** [*group-number*] **timers** [**msec**] *hellotime* [**msec**] *holdtime*	(Optional) Configures the time between hello packets and the time before other routers declare the active Hot Standby or standby router to be down.

	Command or Action	Purpose
	Example: `Router(config-if)# standby 1 timers 1 5`	• *holdtime* --Amount of time the routers take to detect types of failure. A larger hold time means that failure detection will take longer. For the best stability, it is recommended that you set the hold time between 5 and 10 times the hello interval time; otherwise, a failover could falsely occur when no actual failure has happened.
Step 9	**standby delay minimum** [*min-delay*] **reload** [*reload-delay*] **Example:** `Router(config-if)# standby delay minimum 120 reload 120`	Configures the delay period before the initialization of HSRP groups. **Note** It is suggested that you enter 120 as the value for the *reload-delay* argument and leave the *min-delay*argument at the preconfigured default value.
Step 10	Repeat.	Repeat this task on both routers (active and standby) and on both interfaces of each router.

Examples

The following example shows how to configure HSRP on a router:

```
interface Ethernet0/0
 ip address 209.165.201.1 255.255.255.224
 standby 1 ip 209.165.201.3
 standby 1 preempt
 standby 1 name HA-out
 standby 1 track Ethernet1/0
 standby delay minimum 120 reload 120
```

Troubleshooting Tips

To help troubleshoot possible HSRP-related configuration problems, issue any of the following HSRP-related debug commands--**debug standby errors**, **debug standby events**, and **debug standby packets** [**terse**].

What to Do Next

After you have successfully configured HSRP on both the inside and outside interfaces, you should enable SSO as described in the section “Enabling SSO.”

Enabling SSO

Use this task to enable SSO, which is used to transfer Cisco IOS firewall session state information between two routers.

SSO is a method of providing redundancy and synchronization for many Cisco IOS applications and features. SSO is necessary for the Cisco IOS firewall to learn about the redundancy state of the network and to synchronize their internal application state with their redundant peers.

Before You Begin

- You should configure HSRP before enabling SSO.
- To avoid losing SCTP communication between peers, be sure to include the following commands to the local address section of the SCTP section of the IPC configuration:
    - **retransmit-timeout** *retran-min* [*msec*] *retra-max* [*msec*]
    - **path-retransmit** *max-path-retries*
    - **assoc-retransmit** *retries*

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **redundancy inter-device**
4. **scheme standby** *standby-group-name*
5. **exit**
6. **ipc zone default**
7. **association 1**
8. **protocol sctp**
9. **local-port** *local-port-number*
10. **local-ip** *device-real-ip-address* [*device-real-ip-address2*
11. **retransmit-timeout** *retran-min* [*msec*] *retran-max* [*msec*]
12. **path-retransmit** *max-path-retries*
13. **assoc-** retransmit retries
14. **exit**
15. **remote-port** *remote-port-number*
16. **remote-ip** *peer-real-ip-address* [*peer-real-ip-address2*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**redundancy inter-device** **Example:** `Router(config)# redundancy inter-device`	Configures redundancy and enters inter-device configuration mode. To exit inter-device configuration mode, use the **exit** command. To remove all inter-device configuration, use the **no** form of the command.
Step 4	**scheme standby** *standby-group-name* **Example:** `Router(config-red-interdevice)# scheme standby HA-in`	Defines the redundancy scheme that is to be used. Currently, “standby” is the only supported scheme. • *standby-group-name* --Must match the standby name specified in the **standby name** interface configuration command. Also, the standby name should be the same on both routers. **Note** Only the active or standby state of the standby group is used for SSO. The VIP address of the standby group is not required or used by SSO.
Step 5	**exit** **Example:** `Router(config-red-interdevice)# exit`	Exits inter-device configuration mode.
Step 6	**ipc zone default** **Example:** `Router(config)# ipc zone default`	Configures the inter-device communication protocol, Inter-Process Communication (IPC), and enters IPC zone configuration mode. Use this command to initiate the communication link between the active router and standby router.
Step 7	**association 1** **Example:** `Router(config-ipczone)# association 1`	Configures an association between the two devices and enters IPC association configuration mode.
Step 8	**protocol sctp** **Example:** `Router(config-ipczone-assoc)# protocol sctp`	Configures Stream Control Transmission Protocol (SCTP) as the transport protocol and enters SCTP protocol configuration mode.
Step 9	**local-port** *local-port-number* **Example:** `Router(config-ipc-protocol-sctp)# local-port 5000`	Defines the local SCTP port number that is used to communicate with the redundant peer and puts you in IPC transport - SCTP local configuration mode. • *local-port-number* --There is not a default value. This argument must be configured for the local port to enable inter-device redundancy. Valid port values: 1 to 65535. The local port number should be the same as the remote port number on the peer router.

	Command or Action	Purpose
Step 10	**local-ip** *device-real-ip-address* [*device-real-ip-address2* **Example:** `Router(config-ipc-local-sctp)# local-ip 10.0.0.1`	Defines at least one local IP address that is used to communicate with the redundant peer. The local IP addresses must match the remote IP addresses on the peer router. There can be either one or two IP addresses, which must be in the global VRF. A virtual IP address cannot be used.
Step 11	**retransmit-timeout** *retran-min* [*msec*] *retran-max* [*msec*] **Example:** `Router(config-ipc-local-sctp)# retransmit-timeout 300 10000`	Configures the maximum amount of time, in milliseconds, that SCTP will wait before retransmitting data. • *retran-min* : 300 to 60000; default: 300 • *retran-max* : 300 to 60000; default: 600
Step 12	**path-retransmit** *max-path-retries* **Example:** `Router(config-ipc-local-sctp)# path-retransmit 10`	Configures the number of consecutive retransmissions SCTP will perform before failing a path within an association. • *max-path-retries* : 2 to 10; default: 4 retries
Step 13	**assoc-** retransmit retries **Example:** `Router(config-ipc-local-sctp)# assoc -retransmit 10`	Configures the number of consecutive retransmissions SCTP will perform before failing an association. • *retries* : 2 to 10; default: 4 retries
Step 14	**exit** **Example:** `Router(config-ipc-local-sctp)# exit`	Exits IPC transport - SCTP local configuration mode.
Step 15	**remote-port** *remote-port-number* **Example:** `Router(config-ipc-protocol-sctp)# remote-port 5000`	Defines the remote SCTP port number that is used to communicate with the redundant peer and puts you in IPC transport - SCTP remote configuration mode. **Note** *remote-port-number* --There is not a default value. This argument must be configured for the remote port to enable inter-device redundancy. Valid port values: 1 to 65535. The remote port number should be the same as the local port number on the peer router.
Step 16	**remote-ip** *peer-real-ip-address* [*peer-real-ip-address2* **Example:** `Router(config-ipc-remote-sctp)# remote-ip 10.0.0.2`	Defines at least one remote IP address of the redundant peer that is used to communicate with the local device. All remote IP addresses must refer to the same device. A virtual IP address cannot be used.

Examples

The following example shows how to enable SSO:

```
!
redundancy inter-device
 scheme standby HA-in
!
!
ipc zone default
 association 1
  no shutdown
  protocol sctp
   local-port 5000
    local-ip 10.0.0.1
    retransmit-timeout 300 10000
    path-retransmit 10
    assoc-retransmit 10
   remote-port 5000
    remote-ip 10.0.0.2
!
```

Troubleshooting Tips

To help troubleshoot possible SSO-related configuration problems, issue the **debug redundancy** command.

What to Do Next

After you have enabled SSO, you should enable stateful failover for a firewall, as shown in the following section.

Enabling Stateful Failover for a Cisco IOS Firewall

Use this task to enabling Stateful Failover for the Cisco IOS firewall.

Before You Begin

Before performing this task, the Cisco IOS firewall inspect rule must be configured. Also, HSRP and SSO must be configured to enable box-to-box redundancy.

Note The inspect rules should not have ICMP or protocols for which Cisco IOS firewall supports Layer 7 inspection.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** [interface-name]
4. **ip inspect** [**rule**] **in**| **out redundancy stateful** [*hsrp-group-name*]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** [interface-name] **Example:** `Router (config)# interface interface1`	Defines the interface.
Step 4	**ip inspect** [**rule**] **in**| **out redundancy stateful** [*hsrp-group-name*] **Example:** `Router (config)# ip inspect rule1 in/out redundancy stateful group101` **Example:**	Enables stateful failover for this inspect rule. **Note** The hsrp-group-name is the same hsrp-group-name used in the SSO configuration.
Step 5	**exit** **Example:** `Router (config)# exit`	Exit global configuration mode

Configuring the Cisco IOS Firewall HA Update Interval

Use this task to change the amount of time between each update to the standby. The default interval of 10 seconds.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. ip inspect redundancy update seconds [10-60]
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	ip inspect redundancy update seconds [10-60] **Example:** `Router (config)# ip inspect redundancy upate seconds 20` **Example:**	Changes the amount of time between each update to the standby. The default interval of 10 seconds is used if you do not specify a value.
Step 4	**exit** **Example:** `Router (config)# exit`	Exit global configuration mode

Troubleshooting Stateful Failover

The following commands may be used to display information about Stateful Failover messages or sessions. The **debug** commands may be used in any order or independent of the other **debug** commands.

SUMMARY STEPS

1. **enable**
2. **debug ip inspect ha** [**manager** | **update**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ip inspect ha** [**manager** | **update**] **Example:** `Router# debug ip inspect ha manager`	Displays enough information to identify problems with add/delete to ha sessions. • manager (Optional)--Displays in detail the message that the FW HA manager code hands over to CF on the active, and on the standby it displays the message that CF hands over to the FW HA manager. • update (Optional)--Displays updated debug data.

Maintaining Firewall Stateful Failover

The **clear ip inspect ha** commandis usedto clear all inspect ha sessions on the device. If the device is the standby device then it initiates a bulk sync of all session from the active. It is also used to clear the ha statistics on the device

SUMMARY STEPS

1. **enable**
2. **clear ip inspect ha** [**sessions-all** | **statistics**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**clear ip inspect ha** [**sessions-all** | **statistics**] **Example:** `Router# clear ip inspect ha sessions-all all`	The options for this command are: • **sessions-all** --Clears all inspect ha sessions on the device. If the device is the standby device then it initiates a bulk sync of all session from the active. • **statistics** --clears the ha statistics on the device

Displaying Firewall Stateful Failover Information

Use the **show ip inspect ha**{**sessions** [**detail**] | **statistics**} [**vrf** *vrf-name*]}command to display firewall stateful failover information.

SUMMARY STEPS

1. **enable**
2. **show ip inspect ha** {**session** [**detail**] | **statistics**} [**vrf** *vrf-name*]}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip inspect ha** {**session** [**detail**] | **statistics**} [**vrf** *vrf-name*]} **Example:** `Router# show ip inspect ha session`	The options for this command are: • **session** [**detail**]--Displays additional information on pin-holes created for the return traffic, number of bytes that have passed through this session and session time information. • **statistics** --Displays HA sessions statistics for both the Active and Standby devices. • **vrf** *vrf-name* (Optional)--Displays information only for the specified Virtual Routing and Forwarding (VRF) interface.

What to Do Next

The following tables provide examples of Stateful Failover error messages and alert message.

The table below contains the stateful failover HA error messages.

Table 3: Stateful Failover Error Messages

Message	Meaning
`*Apr 13 17:09:20.539: %FW_HA-3-SUBSYS_INIT_FAILED: Firewall High availability subsystem initialization failed`	The HA subsystem initialization failed.
`*Apr 13 16:50:30.007: %FW_HA-3-TW_INIT_FAILED: Firewall High availability update timer initialization failed`	The HA timer wheel initialization failed.

`*Apr 13 16:50:30.007: %FW_HA-3-RF_REG_FAILED: Firewall High availability registration to RF failed`
`*Apr 13 16:50:30.007: %FW_HA-3-CF_REG_FAILED: Firewall High availability registration to CF failed`
	Registration to SSO RF/CF failed.

`May 20 21:57:10.475: %FW_HA-6-NO_INSPECT_RULE_ON_STDBY: Firewall High availability - inspect rule is not configured on standby for interface e0/0 dir in/out`	The Inspect rule is not configured on the standby device.
`*May 20 21:57:10.475: %FW_HA-6-PROT_MISMATCH: Firewall High availability - L4/L7 protocol mismatch between active and standby`	Protocol mismatch between the active and standby devices.
`May 20 21:57:10.475: %FW_HA-6-NO_HSRP_GNAME_ON_STDBY: Firewall High availability - Inpsect redundancy group is not configured on standby for interface e0/0 dir in/out`	The HSRP group is not configured on the standby device.
`*May 20 21:57:10.475: %FW_HA-6-CONFIG_MISMATCH: Firewall High availability - Inspect HA config mismatch between active and standby. act:inspect rule a_test, hsrp_grp a_hsrp_group; stdby:inspect rule s_test hsrp_grp s_hsrp_group`	HA configuration mismatch between the active and standby devices.

If audit trail is configured on the standby HA device the standard alerts that are shown when a session is added or deleted will be changed to reflect that the session is a standby session. The table below contains the stateful failover alert messages.

Table 4: Stateful Failover Alert Messages

Message	Meaning

`*Apr 14 23:53:44.641:`
`%FW-HA-6-SESS_AUDIT_TRAIL_STDBY_START: Start tcp`
`standby session: initiator (10.0.0.10:22955) --`
`responder (11.0.0.10:23)`
	The Standby session is up.

`*Apr 14 23:57:52.891:`
`%FW-HA-6-SESS_AUDIT_TRAIL_STDBY_STOP: Stop tcp standby`
` session: initiator (10.0.0.10:35148) -- responder`
`(11.0.0.10:23)`
	The Standby session is down.

`*Apr 14 23:57:52.891:`
`%FW-HA-6-SESS_AUDIT_TRAIL_STDBY_TO_ACT: Firewall HA`
`transitioning from Standby to Active HA state`
	The device has transitioned from standby to active.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Configuring HSRP	Configuring HSRP

Standards

Standards	Title
No new or modified standards are supported by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported 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
No new or modified RFCs are supported by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Configuration Examples for Stateful Failover

Example Stateful Failover

The following output example shows stateful failover that has been configured on a Cisco IOS router:

```
Router 1)
hostname ha-R1
!
boot-start-marker
boot-end-marker
!
!
redundancy inter-device
scheme standby HAin
!
!
redundancy
logging buffered 10000000 debugging
logging rate-limit console 10000
!
no aaa new-model
!
resource policy
!
!
ipc zone default
association 1
no shutdown
protocol sctp
local-port 5000
local-ip 10.0.0.1
remote-port 5000
remote-ip 10.0.0.2
```

```
!
!
ip inspect tcp idle-time 180
ip inspect name ha-protocols tcp
ip inspect name ha-protocols udp
ip inspect redundancy update seconds 60
!
!
!inside interface
interface Ethernet0/0
ip address 10.0.0.1 255.255.255.0
standby delay minimum 120 reload 120
standby 1 ip 10.0.0.3
standby 1 timers 1 10
standby 1 priority 60
standby 1 preempt
standby 1 name HAin
standby 1 track Ethernet1/0
!
!outside interface
interface Ethernet1/0
ip address 211.0.0.1 255.255.255.0
ip access-group fw-ha-acl in
!! The HSRP group used with the inspect config should be the inside HSRP group
ip inspect ha-protocols out redundancy stateful HAin
standby delay minimum 120 reload 120
standby 2 ip 211.0.0.3
standby 2 timers 1 10
standby 2 priority 60
standby 2 preempt
standby 2 name HAout
standby 2 track Ethernet0/0
!
!
!
! ACL on interface should permit HSRP, HA traffic from active to standby device
ip access-list extended fw-ha-acl
permit ip host 211.0.0.2 host 211.0.0.1
permit ip host 211.0.0.1 host 211.0.0.2
deny any any
!
!
!
!
line con 0
exec-timeout 0 0
line aux 0
##########################################################################
Router 2)
hostname ha-R2
!
boot-start-marker
boot-end-marker
!
!
redundancy inter-device
scheme standby HAin
!
!
redundancy
logging buffered 10000000 debugging
logging rate-limit console 10000
!
no aaa new-model
!
resource policy
!
!
ipc zone default
association 1
no shutdown
protocol sctp
local-port 5000
```

```
local-ip 10.0.0.2
remote-port 5000
remote-ip 10.0.0.1
!
!
ip inspect tcp idle-time 180
ip inspect name ha-protocols tcp
ip inspect name ha-protocols udp
ip inspect redundancy update seconds 60
!
!
!inside interface
interface Ethernet0/0
ip address 10.0.0.2 255.255.255.0
standby delay minimum 120 reload 120
standby 1 ip 10.0.0.3
standby 1 priority 60
standby 1 preempt
standby 1 name HAin
standby 1 track Ethernet1/0
!
!outside interface
interface Ethernet1/0
ip address 211.0.0.2 255.255.255.0
ip access-group fw-ha-acl in
!! The HSRP group used with the inspect config should be the inside HSRP group
ip inspect ha-protocols out redundancy stateful HAin
standby delay minimum 120 reload 120
standby 2 ip 211.0.0.3
standby 2 priority 60
standby 2 preempt
standby 2 name HAout
standby 2 track Ethernet0/0
!
!
!
! ACL on interface should permit HSRP, HA traffic from active to standby device
ip access-list extended fw-ha-acl
permit ip host 211.0.0.2 host 211.0.0.1
permit ip host 211.0.0.1 host 211.0.0.2
!
!
!
!
line con 0
exec-timeout 0 0
line aux 0
```

Feature Information for Cisco IOS Firewall Stateful Failover

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5: Feature Information for Cisco IOS Firewall Stateful Failover

Cisco IOS Firewall Stateful Failover	12.4(6)T	With the introduction of the Stateful Failover, applications and network services are not disrupted if an interface on a router is lost or if a router crashes. With a Stateful Failover configuration, the standby or backup router maintains state information so that firewall operations are maintained in the event of a failure. The following commands are introduced or modified in the feature: **clear ip inspect ha**, **debug ip inspect ha**, **ip inspect**, **show ip inspect**, **show ip inspect ha**.

CHAPTER 4

Configuring Port to Application Mapping

This chapter describes the Cisco IOS Firewall Port to Application Mapping (PAM) feature. PAM enables CBAC-supported applications to be run on nonstandard ports. Using PAM, network administrators can customize access control for specific applications and services to meet the distinct needs of their networks.

For a complete description of the PAM commands in this chapter, refer to the chapter “Port to Application Mapping Commands” of the *Cisco IOS Security Command Reference* . To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the “Identifying Supported Platforms” section in the chapter “Using Cisco IOS Software.”

Information About Port to Application Mapping

Port to Application Mapping (PAM) is a feature of the Cisco IOS Firewall feature set. PAM allows you to customize TCP or UDP port numbers for network services or applications. PAM uses this information to support network environments that run services using ports that are different from the registered or well-known ports associated with an application.

Using the port information, PAM establishes a table of default port-to-application mapping information at the firewall. The information in the PAM table enables Context-based Access Control (CBAC) supported services to run on nonstandard ports. Previously, CBAC was limited to inspecting traffic using only the well-known or registered ports associated with an application. Now, PAM allows network administrators to customize network access control for specific applications and services.

PAM also supports host or subnet specific port mapping, which allows you to apply PAM to a single host or subnet using standard access control lists (ACLs). Host or subnet specific port mapping is done using standard ACLs.

This section contains the following sections:

How PAM Works

PAM generates a table of information that identifies specific applications with specific TCP or UDP port information. When the firewall router first starts up, the PAM table is populated with system-defined mapping information. As you customize the mapping information, the PAM table is modified with the new information. The information in the PAM table serves as the default port mapping for traffic passing through the firewall.

PAM works with CBAC to identify the applications associated with various port numbers, including services running on non-standard ports, as it inspect traffic passing through the firewall. Previously, CBAC was limited to inspecting traffic using only the well-known or registered ports associated with an application.

Entries in the PAM table provide three types of mapping information:

System-Defined Port Mapping

PAM creates a table, or database, of system-defined mapping entries using the well-known or registered port mapping information set up during the system start-up. The system-defined entries comprise all the services supported by CBAC, which requires the system-defined mapping information to function properly. The system-defined mapping information cannot be deleted or changed; that is, you cannot map HTTP services to port 21 (FTP) or FTP services to port 80 (HTTP).

Note You can override the system-defined entries for specific hosts using the PAM host-specific option. Refer to the section "Host-Specific Port Mapping" in this chapter.

The table below lists the default system-defined services and applications in the PAM table.

Table 6: System-Defined Port Mapping

Application Name	Well-Known or Registered Port Number	Protocol Description
cuseeme	7648	CU-SeeMe Protocol
exec	512	Remote Process Execution
ftp	21	File Transfer Protocol (control port)
http	80	Hypertext Transfer Protocol
h323	1720	H.323 Protocol (for example, MS NetMeeting, Intel Video Phone)
login	513	Remote login
mgcp	2427	Media Gateway Control Protocol
msrpc	135	Microsoft Remote Procedure Call

Application Name	Well-Known or Registered Port Number	Protocol Description
netshow	1755	Microsoft NetShow
real-audio-video	7070	RealAudio and RealVideo
rtsp	8559	Real Time Streaming Protocol
shell	514	Remote command
sip	5060	Session Initiation Protocol
smtp	25	Simple Mail Transfer Protocol
sqlnet	1521	SQL-NET
streamworks	1558	StreamWorks Protocol
sunrpc	111	SUN Remote Procedure Call
telnet	23	Telnet
tftp	69	Trivial File Transfer Protocol
vdolive	7000	VDOLive Protocol

User-Defined Port Mapping

Network services or applications that use non-standard ports require user-defined entries in the PAM table. For example, your network might run HTTP services on the non-standard port 8000 instead of on the system-defined default port (port 80). In this case, you can use PAM to map port 8000 with HTTP services. If HTTP services run on other ports, use PAM to create additional port mapping entries. After you define a port mapping, you can overwrite that entry at a later time by simply mapping that specific port with a different application.

Note If you try to map an application to a system-defined port, a message appears that warns you of a mapping conflict.

User-defined port mapping information can also specify a range of ports for an application by establishing a separate entry in the PAM table for each port number in the range.

User-defined entries are saved with the default mapping information when you save the router configuration.

Host-Specific Port Mapping

User-defined entries in the mapping table can include host-specific mapping information, which establishes port mapping information for specific hosts or subnets. In some environments, it might be necessary to override the default port mapping information for a specific host or subnet.

With host-specific port mapping, you can use the same port number for different services on different hosts. This means that you can map port 8000 with HTTP services for one host, while mapping port 8000 with Telnet services for another host.

Host-specific port mapping also allows you to apply PAM to a specific subnet when that subnet runs a service that uses a port number that is different from the port number defined in the default mapping information. For example, hosts on subnet 192.168.21.0 might run HTTP services on non-standard port 8000, while other traffic through the firewall uses the default port for HTTP services, which is port 80.

Host-specific port mapping allows you to override a system-defined entry in the PAM table. For example, if CBAC finds an entry in the PAM table that maps port 25 (the system-defined port for SMTP) with HTTP for a specific host, CBAC identifies port 25 as HTTP protocol traffic on that host.

Note If the host-specific port mapping information is the same as an existing system-defined or user-defined default entries, host-specific port changes have no effect.

PAM and CBAC

CBAC uses the information in the PAM table to identify a service or application from traffic flowing through the firewall. With PAM, CBAC can associate non-standard port numbers with specific protocols. For example, if you use PAM to map port 8000 with HTTP services, CBAC can determine that traffic using port 8000 is an HTTP application.

When to Use PAM

Here are a few examples of when you might want to use PAM:

- Use PAM to apply a non-standard port numbers for a service or application.
- Use PAM when a specific host or subnet uses a port number for an application that is different than the default port number established in the PAM table.
- Use PAM when different hosts use the same port number for different applications.

How to Configure Port to Application Mapping

Configuring Standard ACLs

If you require PAM for a specific host or subnet, use the **access-list** (standard) command in global configuration mode to define an ACL:

Command	Purpose
`Router(config)#` **access-list** *access-list-number* **permit** *source* [*source-wildcard*]	(Optional) Creates a standard ACL that defines the specific host or subnet for host-specific PAM.

Configuring PAM

To configure PAM, use the **ip port-map**command in global configuration mode:

Command	Purpose
`Router(config)#` **ip port-map** *appl-name* **port** *port-num* [**list** *acl-num*]	Establishes a port mapping entry using the TCP or UDP port number and the application name. (Optional) Use the list option to associate this port mapping to the specific hosts in the ACL. (PAM uses standard access lists only.) If an access list is included, the hosts defined in that ACL have the application *appl-name* running on port *port-num*.

Verifying PAM

To verify the port mapping information, enter the **show ip port-map** command in privileged EXEC mode and review the entries:

```
Router# show ip port-map
```

This command displays all entries in the PAM table, including the system-defined entries.

For PAM configuration examples using the commands in this chapter, refer to the "Configuration Examples for Port to Application Mapping" section at the end of this chapter.

Monitoring and Maintaining PAM

The following commands can be used to monitor and maintain PAM:

Command	Purpose
`Router#` **show ip port-map** `[`*appl-name* `|` **port** *port-num*`]`	Displays the port mapping information, including the system-defined entries. Include the application name to display a list of entries by application. Include the port number to display the entries by port.
`Router(config)#` **no ip port-map** *appl-name* **port** *port-num* `[`**list** *acl-num*`]`	Deletes user-defined port mapping information. This command has no effect on the system-defined port mapping information.

Configuration Examples for Port to Application Mapping

Example Mapping an Application to a Non-Standard Port

In this example, non-standard port 8000 is established as the user-defined default port mapping for HTTP services:

```
ip port-map http port 8000
```

Example Mapping an Application with a Port Range

The following PAM entries establish a range of non-standard ports for HTTP services:

```
ip port-map http 8001
ip port-map http 8002
ip port-map http 8003
ip port-map http 8004
```

Example Invalid Port Mapping Entry

This example is not valid because it tries to establish port 21, which is the system-defined default port for FTP, as the user-defined port for HTTP services:

```
ip port-map http port 21
```

Example Mapping an Application to a Port for a Specific Host

In this example, a specific host uses port 8000 for FTP services. ACL 10 identifies the server address (192.168.32.43), while port 8000 is mapped with FTP services.

```
access-list 10 permit 192.168.32.43
ip port-map ftp port 8000 list 10
```

Example Mapping an Application to a Port for a Subnet

In this example, a specific subnet runs HTTP services on port 8080. ACL 50 identifies the subnet, while port 8080 is mapped with HTTP services.

```
access-list 50 permit 192.168.92.0 0.0.0.255
ip port-map http 8080 list 50
```

Example Overriding a System-Defined Port Mapping

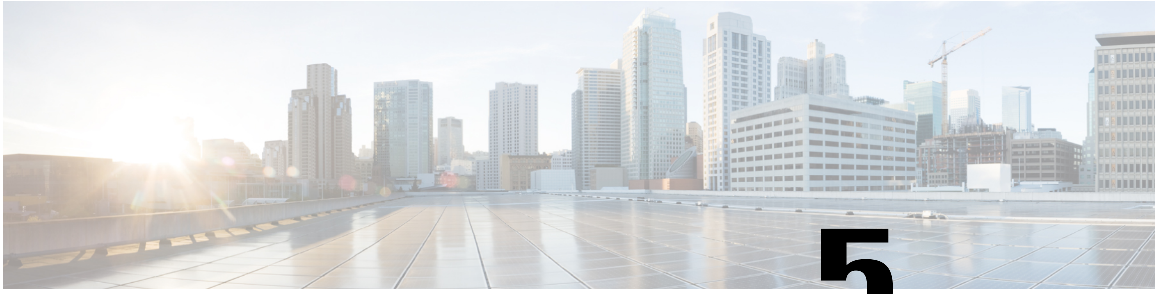
In this example, a specific host runs HTTP services on port 25, which is the system-defined port number for SMTP services. This requires a host-specific PAM entry that overrides the system-defined default port mapping for HTTP, which is port 80. ACL 15 identifies the host address (192.168.33.33), while port 25 is mapped with HTTP services.

```
access-list 15 permit 192.168.33.33
ip port-map http port 25 list 15
```

Example Mapping Different Applications to the Same Port

In this example, the same port number is required by different services running on different hosts. Port 8000 is required for HTTP services for host 192.168.3.4, while port 8000 is also required for FTP services for host 192.168.5.6. ACL 10 and ACL 20 identify the specific hosts, while the PAM entries map the ports with the services for each ACL.

```
access-list 10 permit 192.168.3.4
access-list 20 permit 192.168.5.6
ip port-map http port 8000 list 10
ip port-map http ftp 8000 list 20
```



CHAPTER 5

Cisco IOS Firewall MIB

The Cisco IOS Firewall MIB feature introduces support for the Cisco Unified Firewall MIB, which helps to manage and monitor firewall performance via Simple Network Management Protocol (SNMP). Statistics can be collected and monitored via standards-based SNMP techniques for firewall features such as stateful packet inspection and URL filtering.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites Cisco IOS Firewall MIB

Before you can provide firewall connection and URL filtering statistics via SNMP, you must set up the firewall by performing the following tasks:

- Configure a firewall policy via the **ip inspect name** command.

- Enable the firewall by applying the firewall on a target via the **interface** command followed by the **ip inspect** command.
- Enable URL filtering, if applicable, via the **ip urlfilter server vendor** command.

You must also enable SNMP on the router. For more information on enabling SNMP, see the section "Enabling SNMP for Firewall Sessions" later in this document.

Restrictions for Cisco IOS Firewall MIB

- Cisco does not support all of the MIB variables that are defined in the Cisco Unified Firewall MIB. For a list of variables that are supported by this feature, see the table below.
- MIB statistics are not provided when the firewall is configured using CPL.

Memory and Performance Impact

Depending on the number of targets that have a configured firewall and the number of configured URL filtering servers, the MIB functionality can create an adverse impact on memory. For each firewall policy that is configured on your system, more memory is required to store SNMP statistics.

The following information defines the minimum memory requirements for connection statistics only:

- Global connection statistics: approximately 64 bytes.
- Protocol-specific statistics: multiply the number of configured protocols by 56 to determine the minimum memory requirement.
- Policy-target-protocol statistics: multiply the number of configured protocols and the number of targets for which the firewall policies are configured by 48 to determine the minimum memory requirement.

The following information defines the minimum memory requirements for URL filtering statistics only:

- Global URL filtering statistics: approximately 96 bytes.
- URL filtering server-specific statistics: multiply the number of configured URL filtering servers by 40 to determine the minimum memory requirement.

Information About Cisco IOS Firewall MIB

Connection Statistics

Connection statistics are a record of the firewall traffic streams that have attempted to flow through the firewall system. Connection statistics can be displayed on a global basis (that is, an aggregate of all connection statistics for the entire router), protocol-specific basis, or a firewall-policy-specific basis. The Firewall can allow, drop, or deny the connection based on firewall policies and firewall resources.

The table below lists all supported connection statistics--global, protocol-specific[4], or firewall-policy-specific[5]--that are available via SNMP.

Table 7: Connection Statistics

Statistic Type	Connection Type	Description
• Global • Protocol-specific • Firewall-policy-specific	Aborted	Number of connections that were abnormally terminated after successful establishment
• Global • Protocol-specific • Firewall-policy-specific	Active	Number of connections that are currently active
• Global • Protocol-specific • Firewall-policy-specific	Attempted	Number of connection attempts sent to the firewall system
Global	Embryonic	Number of embryonic-application-layer connections
Global	Expired	Number of connections that were active but have since been terminated normally
• Global • Protocol-specific	Five-Minute Connection Rate	Number of connection attempts that were established per second, averaged over the last 300 seconds
• Global • Protocol-specific • Firewall-policy-specific	Half-Open	Number of connections that are currently in the process of being established (half-open)

[4] All protocol-based statistics can be accessed with the following index--protocol, which is the protocol of interest such as ICMP, UDP, TCP, HTTP, and FTP. The protocols, which are a predefined static list, must be specified

[5] All firewall-policy-specific statistics can be accessed with the following indexes: Policy, which is the name of the firewall security policy of interest. (The policy name is specified via the ip inspect name command.) Policy target type, which is the type of physical or virtual target that has the policy name applied to it. Currently, only include interface targets are supported.

Statistic Type	Connection Type	Description
• Global • Protocol-specific	One-Minute Connection Rate	Number of connection attempts that were establish per second, averaged over the last 60 seconds
• Global • Protocol-specific • Firewall-policy-specific	Policy Declined	Number of connection attempts that were declined due to application of a firewall security policy
• Global • Protocol-specific • Firewall-policy-specific	Resource Declined	Number of connection attempts that were declined due to firewall resource constraints

URL Filtering Statistics

URL Filtering feature provides an Internet management application that allows you to control web traffic for a given host or user on the basis of a specified security policy. URL filtering statistics include the status of distinct URL filtering servers that are configured on the firewall and the impact of the performance of the URL filtering servers on the latency and throughput of the firewall.

The tables below list all supported URL filtering statistics--on a global basis or per server--that are available via SNMP.

Table 8: Global URL Filtering Statistics (across all servers)

Connection Type	Description
Five minute URL Filtering Requests Declined Rate	Rate at which URL access requests were declined by the firewall via the URL filtering server or the firewall exclusive domain configuration, averaged over the last 300 seconds.
Five minute URL Filtering Requests Resource Dropped Rate	Rate at which URL access requests were dropped by the firewall due to firewall resource constraints, averaged over the last 300 seconds.
One minute URL Filtering Requests Declined Rate	Rate at which URL access requests were declined by the firewall via the URL filtering server or the firewall exclusive domain configuration, averaged over the last 60 seconds.

Connection Type	Description
One minute URL Filtering Requests Resource Dropped Rate	Rate at which URL access requests were dropped by the firewall due to firewall resource constraints, averaged over the last 60 seconds.
URL Filtering Allow Mode On	Displays whether the firewall has allowed or discarded URL requests when the URL filtering server is not available. Returns a “true” statistics if the firewall allows all requested URLs to be retrieved from the remote host when the URL server is not available; returns a “false” statistic of the firewall discards all URL.
URL Filtering Allow Mode Requests Allowed	Number of URL access requests that were allowed by the firewall when the URL filtering server was not available.
URL Filtering Allow Mode Requests Denied	Number of URL access requests that were denied by the firewall when the URL filtering server was not available.
URL Filtering Enabled	Displays whether or not URL filtering is enabled. Returns a “false” statistic if the firewall will not perform URL filtering, even if the system contains configuration information that pertains to other aspects of URL filtering.
URL Filtering Late Responses	Number of responses from the URL filtering server that were received after the original URL access request was dropped by the Firewall.
URL Filtering Requests Allowed	Number of URL access requests allowed by the firewall via the use of the URL filtering server or the firewall exclusive domain configuration.
URL Filtering Requests Declined	Number of URL access requests that were declined by the firewall via the URL filtering server or the firewall exclusive domain configuration.
URL Filtering Requests Processed	Number of URL access requests that were processed by the firewall.
URL Filtering Request Process Rate	Number of URL access requests that were processed per second by the firewall, averaged over the last 300 seconds.
URL Filtering Requests Resource Dropped	Number of incoming URL access requests that were dropped by the Firewall due to firewall resource constraints.

Connection Type	Description
URL Filtering Responses Resource Dropped	Number of responses to URL access requests from remote hosts that were dropped by the firewall due to resource constraints while the firewall was waiting for a response from the URL filtering server.
URL Filtering Server Timeouts	Number of times the firewall did not receive a response from the URL Filtering server.

Table 9: Per server URL Filtering Statistics

Connection Type	Description
URL Filtering Protocol Version	Version of the transport protocol that is used by the firewall to communicate with the URL filtering server. For TCP, valid version values are 1 and 4. For UDP, 1 is the only valid version.
URL Filtering Server Late Responses	Number of URL access responses received by the firewall from the URL filtering server after the original URL access request was dropped by the firewall.
URL Filtering Server Requests	Number of URL access requests forwarded by the firewall to the URL filtering server.
URL Filtering Server Requests Allowed	Number of URL access requests allowed by the URL filtering server. The count does not include late responses.
URL Filtering Server Requests Declined	Number of URL access requests declined by the URL filtering server. The count does not include late responses.
URL Filtering Server Responses	Number of URL access responses received by the firewall from the URL filtering server. The count does not include late responses.
URL Filtering Server Response Time Rate	Average round-trip response time of the URL filtering server, averaged over the last 300 seconds. A value of zero indicates that there was insufficient data to compute this value over the last time interval.
URL Filtering Server Status	Status of the URL filtering server: ONLINE or OFFLINE.
URL Filtering Server Timeouts	Number of times the URL filtering server failed to respond to URL access requests sent by the firewall.

Connection Type	Description
URL Filtering Server Transport Protocol	Transport protocol that is used by the firewall to communicate with the URL filtering server. The protocol will be TCP, UDP, or DEFAULT. DEFAULT is used in implementations that do not explicitly specify a transport protocol.
URL Filtering Server Vendor	Vendor who provided the URL filtering server. Currently only Websense and N2H2 servers are supported.

A URL filtering server is identified by the following items, which also form the indexes into the URL filtering server statistics table:

- URL Filtering Server Address Type--Type of IP address of the URL filtering server. For example, IPv4 or IPv6.
- URL Filtering Server Address--IP address of the URL filtering server.
- URL Filtering Server Port--Port number that the URL filtering server uses to receive filtering requests.

Firewall MIB Traps

To receive firewall MIB traps, you need a management station, and you must enable the **snmp-server enable trap firewall serverstatuschange** command (as shown in the configuration task table below).

Output for the SNMP trap fields, which are displated in on the management station, are as follows:

- Server IP Address Type (IPv4 or IPv6)
- Server IP Address Type Length. (4 for IPv4 and 16 for IPv6)
- Server IP Address
- Server Port

Note Only IPv4 is currently supported.

How to Configure Cisco IOS Firewall MIB

Enabling SNMP for Firewall Sessions

Perform this task to enable SNMP for firewall-related session management.

Before You Begin

Before you can begin monitoring firewall performance via SNMP, you must set up the firewall by performing the following tasks:

- Configure a firewall policy via the **ip inspect name** command.

Note Statistics are collected only for protocols that are specified via the **ip inspect name** command.

- Enable the firewall by applying the firewall on a target via the **interface** command followed by the **ip inspect** command.
- Enable URL filtering, if applicable, via the **ip urlfilter server vendor** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server community** *string*
4. **snmp-server host** *hostname community-string*
5. **snmp-server enable traps firewall** [**serverstatuschange**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server community** *string* **Example:** `Router(config)# snmp-server community public`	Sets up the community access string to permit access to the SNMP.

	Command or Action	Purpose
Step 4	**snmp-server host** *hostname* *community-string* **Example:** `Router(config)# snmp-server host 192.168.1.1 version 2c public`	Specifies the recipient of the firewall-related SNMP notifications.
Step 5	**snmp-server enable traps firewall** [**serverstatuschange** **Example:** `Router(config)#  snmp-server enable traps firewall serverstatuschange`	Enables firewall-related SNMP notifications.

What to Do Next

After the firewall and SNMP have been properly enabled, statistics will begin to accumulate after the traffic flow starts. To verify whether statistics are being collected and view MIB counters, you can perform at least one of the steps in the task “Verifying Firewall Connection and URL Filtering Statistics.”

Verifying Firewall Connection and URL Filtering Statistics

Use this task to verify firewall connection and URL filtering statistics via command-line interface (CLI). (These statistics can also be collected via any SNMP-capable client.)

Note Effective with Cisco IOS Release 12.4(20)T, the **debug ip inspect** command is replaced by the **debug policy-firewall** command. See the Cisco IOS Debug Command Reference for more information.

SUMMARY STEPS

1. **enable**
2. show ip inspect mib connection-statistics {global | l4-protocol {all | icmp | tcp | udp} | l7-protocol {all | other | telnet | ftp} | policy policy-name target target name {l4-protocol {all | icmp | tcp | udp} | l7-protocol {all | other | telnet | ftp}}
3. show ip urlfilter [mib] statistics [{global | server {ip-address [port] | all}}]
4. debug ip inspect mib {object-creation | object-deletion | events | retrieval | update}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	show ip inspect mib connection-statistics {global | l4-protocol {all | icmp | tcp | udp} | l7-protocol {all | other | telnet | ftp} | policy policy-name target target name {l4-protocol {all | icmp | tcp | udp} | l7-protocol {all | other | telnet | ftp}} **Example:** `Router# show ip inspect mib connection-statistics global`	Displays firewall performance summary statistics that are monitored via SNMP. • **global** --Provides global connection statistics. • **l4-protocol** --Provides Layer 4 statistics for a specified protocol. • **l7-protocol** --Provides Layer 7 statistics for a specified protocol. • **policy** *policy-name* **target** *target-name* --Provides statistics on a per-policy target basis. For example, per firewall policy name and the interface on which the firewall is configured.
Step 3	show ip urlfilter [mib] statistics [{global | server {ip-address [port] | all}}] **Example:** `Router# show ip urlfilter mib statistics global`	Displays URL filtering statistics for firewall-related MIB events.
Step 4	debug ip inspect mib {object-creation | object-deletion | events | retrieval | update} **Example:** `Router# debug ip inspect mib events`	Displays messages about firewall MIB events.

Troubleshooting Tips

All statistics are accumulated since the last reboot of the firewall system. Thus, you must reboot the system to clear MIB connection statistics from your system.

Configuration Examples for Cisco IOS Firewall MIB Monitoring

Example Sample Cisco IOS Firewall Configuration

The following output from the **show running-config** command shows how to configure a Cisco IOS Firewall:

```
Router# show running-config
Building configuration...
Current configuration : 2205 bytes
!
version 12.4
no service pad
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service internal
!
hostname Router
!
boot-start-marker
boot-end-marker
!
no logging console
!
no aaa new-model
!
resource policy
!
clock timezone MST -8
clock summer-time MDT recurring
no ip cef
!
!
!
!
ip inspect name test tcp
ip inspect name test udp
ip inspect name test icmp timeout 30
ip inspect name test ftp
ip inspect name test http
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
policy-map ratelimit
class class-default
police cir 10000000
conform-action transmit
exceed-action drop
!
!
!
!
```

```
!
!
!
interface FastEthernet0/0
ip address 192.168.27.2 255.255.255.0
ip access-group 101 out
ip inspect test in
duplex full
service-policy input ratelimit
!
interface FastEthernet1/0
no ip address
no ip route-cache
shutdown
duplex half
!
interface FastEthernet4/0
ip address 192.168.127.2 255.255.255.0
ip access-group 102 in
duplex full
service-policy input ratelimit
!
router eigrp 100
network 192.168.27.0
network 192.168.127.0
no auto-summary
no eigrp log-neighbor-changes
no eigrp log-neighbor-warnings
!
ip default-gateway 192.168.27.116
ip route 192.168.100.0 255.255.255.0 192.168.27.1
ip route 192.168.200.0 255.255.255.0 192.168.127.1
no ip http server
no ip http secure-server
!
!
!
logging alarm informational
access-list 101 permit tcp any any fragments
access-list 101 permit udp any any fragments
access-list 101 deny tcp any any
access-list 101 deny udp any any
access-list 101 permit ip any any
access-list 102 permit tcp any any fragments
access-list 102 permit udp any any fragments
access-list 102 permit udp any gt 1024 any eq snmp
access-list 102 deny tcp any any
access-list 102 deny udp any any
access-list 102 permit ip any any
snmp-server community public RO
snmp-server location FW Testbed UUT
snmp-server contact STG/IOS FW Devtest
!
!
!
!
!
!
control-plane
!
!
!
!
!
!
gatekeeper
shutdown
!
!
line con 0
exec-timeout 0 0
stopbits 1
line aux 0
```

```
stopbits 1
line vty 0 4
login
!
exception core-file sisu-devtest/coredump/Router.core
exception dump 192.168.27.116
!
end
```

Example Sample URL Filtering Configuration

The following sample output from the **show running-config** command shows how to configure a Websense server for URL filtering:

```
Router# show running-config


Building configuration...
Current configuration : 2043 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service internal
!
hostname Router
!
boot-start-marker
boot-end-marker
!
no logging console
!
no aaa new-model
!
resource policy
!
clock timezone MST -8
clock summer-time MDT recurring
no ip cef
!
!
ip inspect name test tcp
ip inspect name test udp
ip inspect name test http urlfilter
!
!
ip urlfilter allow-mode on
ip urlfilter exclusive-domain deny www.cnn.com
ip urlfilter exclusive-domain permit www.cpp.com
ip urlfilter server vendor websense 192.168.29.116
!
!
!
!
!
!
!
!
!
interface FastEthernet0/0
ip address 192.168.29.2 255.255.255.0
ip access-group 101 out
ip inspect test in
speed auto
full-duplex
!
interface FastEthernet0/1
```

```
ip address 192.168.129.2 255.255.255.0
ip access-group 102 in
duplex auto
speed auto
!
router eigrp 100
network 192.168.29.0
network 192.168.129.0
no auto-summary
no eigrp log-neighbor-changes
no eigrp log-neighbor-warnings
!
ip default-gateway 192.168.28.116
ip route 192.168.100.0 255.255.255.0 192.168.29.1
ip route 192.168.200.0 255.255.255.0 192.168.129.1
!
!
ip http server
no ip http secure-server
!
access-list 101 permit tcp any any fragments
access-list 101 permit udp any any fragments
access-list 101 deny tcp any any
access-list 101 deny udp any any
access-list 101 permit ip any any
access-list 102 permit tcp any any fragments
access-list 102 permit udp any any fragments
access-list 102 permit udp any gt 1024 any eq snmp
access-list 102 deny tcp any any
access-list 102 deny udp any any
access-list 102 permit ip any any
snmp-server community public RO
snmp-server location FW Testbed UUT
snmp-server contact STG/IOS FW Devtest
!
!
!
!
control-plane
!
!
!
line con 0
exec-timeout 0 0
transport output all
line aux 0
transport output all
line vty 0 4
login
!
exception core-file sisu-devtest/coredump/Router.core
exception dump 192.168.28.116
!
webvpn context Default_context
ssl authenticate verify all
!
no inservice
!
!
end
```

Example show ip inspect mib Output

The following examples are sample outputs from the **show ip inspect mib**command with global or protocol-specific keywords:

Global MIB Statistics

```
Router# show ip inspect mib connection-statistics global

--------------------------------------------------
Connections Attempted 7
Connections Setup Aborted 0
Connections Policy Declined 0
Connections Resource Declined 0
Connections Half Open 2 Connections Active 3
Connections Expired 2
Connections Aborted 0
Connections Embryonic 0
Connections 1-min Setup Rate 5
Connections 5-min Setup Rate 7
```

Protocol-Based MIB Statistics

```
Router# show ip inspect mib connection-statistics l4-protocol tcp


--------------------------------------------------
Protocol tcp
Connections Attempted 3
Connections Setup Aborted 0
Connections Policy Declined 0
Connections Resource Declined 0
Connections Half Open 1
Connections Active 2
Connections Aborted 0
Connections 1-min Setup Rate 3
Connections 5-min Setup Rate 3
Router# show ip inspect mib connection-statistics l7-protocol http


--------------------------------------------------
Protocol http
Connections Attempted 3
Connections Setup Aborted 0
Connections Policy Declined 2
Connections Resource Declined 0
Connections Half Open 0
Connections Active 1
Connections Aborted 0
Connections 1-min Setup Rate 1
Connections 5-min Setup Rate 2
```

Policy-Target-Based MIB Statistics

```
Router# show ip inspect mib connection-statistics policy ftp interface GigabitEthernet0/0
l4-protocol tcp


! Policy Target Protocol Based Connection Summary Stats
------------------------------------------------------
Policy ftp-inspection
Target GigabitEthernet0/0
Protocol tcp
Connections Attempted 3
Connections Setup Aborted 0
Connections Policy Declined 0
Connections Resource Declined 0
Connections Half Open 1
Connections Active 2
Connections Aborted 0
Router# show ip inspect mib connection-statistics policy ftp interface GigabitEthernet0/0
l7-protocol ftp
```

```
! Policy Target Protocol Based Connection Summary Stats
------------------------------------------------------
Policy ftp-inspection
Target GigabitEthernet0/0
Protocol ftp
Connections Attempted 3
Connections Setup Aborted 0
Connections Policy Declined 0
Connections Resource Declined 0
Connections Half Open 1
Connections Active 2
Connections Aborted 0
```

Example show ip urlfilter mib statistics command output

The following example is sample output when MIBs are enabled to track URL filtering statistics across the entire device (global):

```
Router# show ip urlfilter mib statistics global

URL Filtering Group Summary Statistics
------------------------------------------------------
URL Filtering Enabled
Requests Processed 260
Requests Processed 1-minute Rate 240
Requests Processed 5-minute Rate 215
Requests Allowed 230
Requests Denied 30
Requests Denied 1-minute Rate 15
Requests Denied 5-minute Rate 0
Requests Cache Allowed 5
Requests Cache Denied 5
Allow Mode Requests Allowed 15
Allow Mode Requests Denied 15
Requests Resource Dropped 0
Requests Resource Dropped 1-minute Rate 0
Requests Resource Dropped 5-minute Rate 0
Server Timeouts 0
Server Retries 0
Late Server Responses 0
Access Responses Resource Dropped 0
```

The following example is sample output when MIBs are enabled to track URL filtering statistics across the server with IP address 192.168.27.116:

```
Router# show ip urlfilter mib statistics server address 192.168.27.116
URL Filtering Server Statistics
------------------------------------------------------
URL Server Host Name 192.168.27.116
Server Address 192.168.27.116
Server Port 15868
Server Vendor Websense
Server Status Online
Requests Processed 4
Requests Allowed 1
Requests Denied 3
Server Timeouts 0
Server Retries 9
Responses Received 1
Late Server Responses 12
1 Minute Average Response Time 0
5 Minute Average Response Time 0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	*Cisco IOS Security Command Reference*
Description of SNMP, SNMP MIBs, and how to configure SNMP on Cisco devices	“Configuring SNMP Support”
Description of Cisco IOS firewalls and functions such as how to configure a firewall and URL filtering	“Configuring Context-based Access Control”

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
• CISCO-UNIFIED-FIREWALL-MIB.my • CISCO-FIREWALL-TC.my	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

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

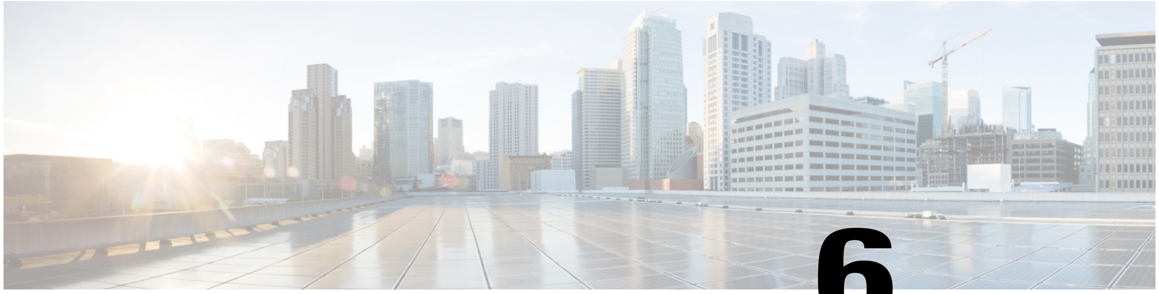
Feature Information for Cisco IOS Firewall MIB

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 10: Feature Information for Cisco IOS Firewall MIB

Feature Name	Releases	Feature Information
Cisco IOS Firewall MIB	12.4(6)T	Introduces support for the Cisco Unified Firewall MIB, which helps to manage and monitor firewall performance via SNMP. Statistics can be collected and monitored via standards-based SNMP techniques for firewall features such as stateful packet inspection and URL filtering. The following commands were introduced or modified: **debug ip inspect**, **show ip inspect**, **show ip urlfilter statistics**, **snmp-server enable traps firewall**.



CHAPTER 6

Cisco IOS Firewall Performance Improvements

The Cisco IOS Firewall Performance Improvements feature introduces three performance metrics for Context-Based Access Control (CBAC)-- Throughput Improvement, Connections Per Second Improvement, and CPU Utilization Improvement.

CBAC is a context-based firewall that performs the following:

- Inspects traffic in one direction for network, transport, and application layer information
- Extracts relevant port information
- Dynamically creates access list entries for return traffic
- Closes ports at the end of a connection

CBAC also forces protocol conformance for some protocols, has a limited vulnerability signature detection mechanism, and extensive denial-of-service (DOS) prevention mechanisms. However, many of these features are CPU intensive, thereby, adversely affecting the performance of the router. The router is also affected during times of heavy traffic due to the processing of the base engine itself. With this feature, the performance of the router running CBAC is no longer subdued.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Cisco IOS Firewall Performance Improvements

To benefit from the performance enhancements, your router must be running CBAC.

Information About Cisco IOS Firewall Performance Improvements

Throughput Improvement

Throughput is a metric defined by the number of packets transferred from the input interface to the output interface within 1 second by a router running CBAC. When the CBAC base engine inspects packets that belong to an existing session, it must find out which session the packet belongs to; thus, the base engine implements a hash table to search for the session of the packet.

Collisions in a hash table result in poor hash function distribution because many entries are hashed into the same bucket for certain patterns of addresses. Even if a hash function distribution evenly dispenses the input across all of the buckets, a small hashtable size will not scale well if there are a large number of sessions. As the number of sessions increase, the collisions increase, which increases the length of the linked lists, thereby, deteriorating the throughput performance.

The Cisco IOS Firewall Performance Improvements feature allows users to dynamically change the size of the session hash table without reloading the router by using the **ip inspect hashtable** command. By increasing the size of the hash table, the number of sessions per hash bucket can be reduced, which improves the throughput performance of the base engine.

Connections Per Second Improvement

Connections per second is a metric defined by the number of short-lived connections that can be created and deleted within 1 second by a router running CBAC. (These connections apply only to TCP connections because UDP is a connectionless protocol.)

Initially, CBAC had several restrictions that limited the connections per second metric. While a packet was being processed for connection setup and connection teardown of TCP connections, the base engine (which allocates and de-allocates memory while processing packets) would “bump up” several packets to the process switching path. Bumping up these packets drastically slowed down their processing. Also, the base engine had to process each packet again when it was bumped up into the process switching path, which also contributed to the degrading performance.

The Cisco IOS Firewall Performance Improvements feature prevents these restrictions by allowing only the first packet of any connection to be bumped up to the process switching path while the remaining packets are processed by the base engine in the fast path. Thus, the base engine is no longer slowed down by bumping up several packets or by processing packets twice.

Note In this document, a connection is defined as creating a session, sending a data packet, and immediately deleting a session.

CPU Utilization Improvement

The CPU utilization of the router running CBAC can be measured while a specific throughput or connections per second metric is maintained. This improvement is used in conjunction with the throughput and connections per second metrics.

Benefits

Layer 4 Processing Performance Improvement

This enhancement improves the connections per second metric and the CPU utilization. The code path for connection initiation and teardown was rewritten, thereby, enabling quicker creation of the connections per second metric, which reduces CPU utilization per connection.

Hash Table Function Performance Improvement

With this enhancement, the hash function has been rewritten to ensure better distribution. This newly improved feature allows users to dynamically configure the size of the session hash table from 1K to 8K. When a packet belonging to an existing session comes into the router, a hash table is used to map the packet to an existing firewall session. As the number of sessions increases, the number of sessions hashing into the same bucket increases if the size of the hash table is fixed. By allowing the user to change the size of the hash table when the number of concurrent sessions increases or to reduce the search time for the session, the throughput performance of the base engine is greatly improved.

Application Module Tuning Performance Improvement

This enhancement makes changes to application modules, ensuring that only the connection-initiating packet from all the packets belonging to the connection initiation and teardown is bumped up to the process switching path. Thus, the connections per second metric is significantly improved.

How to Configure Cisco IOS Firewall Performance Improvements

See the following sections for configuration tasks for the Cisco IOS Firewall Performance Improvements feature. Each task in the list is identified as either required or optional.

Changing the Size of the Hash Table

You can increase the hash table to improve packet distribution. To change the size of the session hash table, use the following command in global configuration mode:

Command	Purpose
`Router#` **ip inspect hashtable** *number*	Changes the size of the hash table. • The *number* argument specifies the size of the hash table in terms of buckets. Possible values for the hash table are 1024, 2048, 4096, and 8192; the default value is 1024. **Note** You should increase the hash table size when the total number of sessions running through the CBAC router is approximately twice the current hash size; decrease the hash table size when the total number of sessions is reduced to approximately half the current hash size. Essentially, try to maintain a 1:1 ratio between the number of sessions and the size of the hash table.

Verifying CBAC Configurations

To verify all CBAC configurations and all existing sessions that are currently being tracked and inspected by CBAC, use the **show ip inspect all**command in EXEC mode.

Configuration Examples for Cisco IOS Firewall Performance Improvements

Example Changing the Size of the Hash Table

The following example shows how to change the size of the hash table to 2048 buckets:

```
ip inspect hashtable 2048
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	*Cisco IOS Security Command Reference*

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Cisco IOS Firewall Performance Improvements

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 11: Feature Information for Cisco IOS Firewall Performance Improvements

Feature Name	Releases	Feature Information
Cisco IOS Firewall Performance Improvements\	12.2(8)T	The Cisco IOS Firewall Performance Improvements feature introduces three performance metrics for Context-Based Access Control (CBAC)--Throughput Improvement, Connections Per Second Improvement, and CPU Utilization Improvement. The following commands were introduced or modified: **ip inspect hashtable**.

CHAPTER 7

Cisco IOS Firewall Support for TRP

To guarantee service and security, the deployment of voice services over IP networks requires special handling of secondary channels within the network. When Trust Relay Points (TRPs) are implemented in voice networks, the networks must account for the following caveats when handling the opening of secondary channels.

- Networks do not always see the signaling messages. (The signaling messages are most likely encrypted.)
- Networks that do see signaling messages cannot deep inspect the messages.
- Networks use other means to learn about the media channels that are being negotiated and opened.

Consequently, transparent entities, such as the Cisco IOS Firewall, that are operating on the networks, must process media channels differently.

This feature enables Cisco IOS Firewall to process Session Traversal Utilities for NAT (STUN). STUN messages open connections between ports for secondary channels, known as pinholes, which are necessary for implementation of TRPs in voice networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Firewall Support for TRP

Before configuring STUN to open pinholes for data, ensure that the voice protocol control packets in your network are not blocked by the Cisco IOS Firewall.

Restrictions for Firewall Support for TRP

- You must configure different agent IDs under a single parameter map. If different agent IDs are configured under two different parameter maps and then the STUN inspection of the two parameter maps are out in the same policy map (per the sample configuration below), the firewall will drop the packet. For example, if you are sending a packet with agent ID 21, the firewall will check the first class map called “stun-ice” and then drop the packet because it did not find a match in that class map.

```
parameter-map type protocol-info stun-ice cfd1
 authorization agent-id 20 shared-secret 12345flower12345 cat-window 15
 authorization agent-id 22 shared-secret 12345cisco54321 cat-window 15
parameter-map type protocol-info stun-ice cfd2
 authorization agent-id 21 shared-secret 12345flower54321 cat-window 15
!
class-map type inspect match-all stun-ice
 match protocol stun-ice cfd1
class-map type inspect match-any stun-ice1
 match protocol stun-ice cfd2
!
policy-map type inspect policy_test
 class type inspect class_1
  pass
 class type inspect sip_ctrl_channel
  inspect
 class type inspect stun-ice
  inspect
 class type inspect stun-ice1
  inspect
 class class-default
  drop
```

Information About Firewall Support for TRP

Cisco IOS Firewall

The Cisco IOS Firewall extends the concept of static access control lists (ACLs) by introducing dynamic ACL entries that open on the basis of the necessary application ports on a specific application and close these ports at the end of the application session. The Cisco IOS Firewall achieves this functionality by inspecting the application data, checking for conformance of the application protocol, extracting the relevant port information to create the dynamic ACL entries, and closing these ports at the end of the session. The Cisco IOS Firewall is designed to easily allow a new application inspection whenever support is needed.

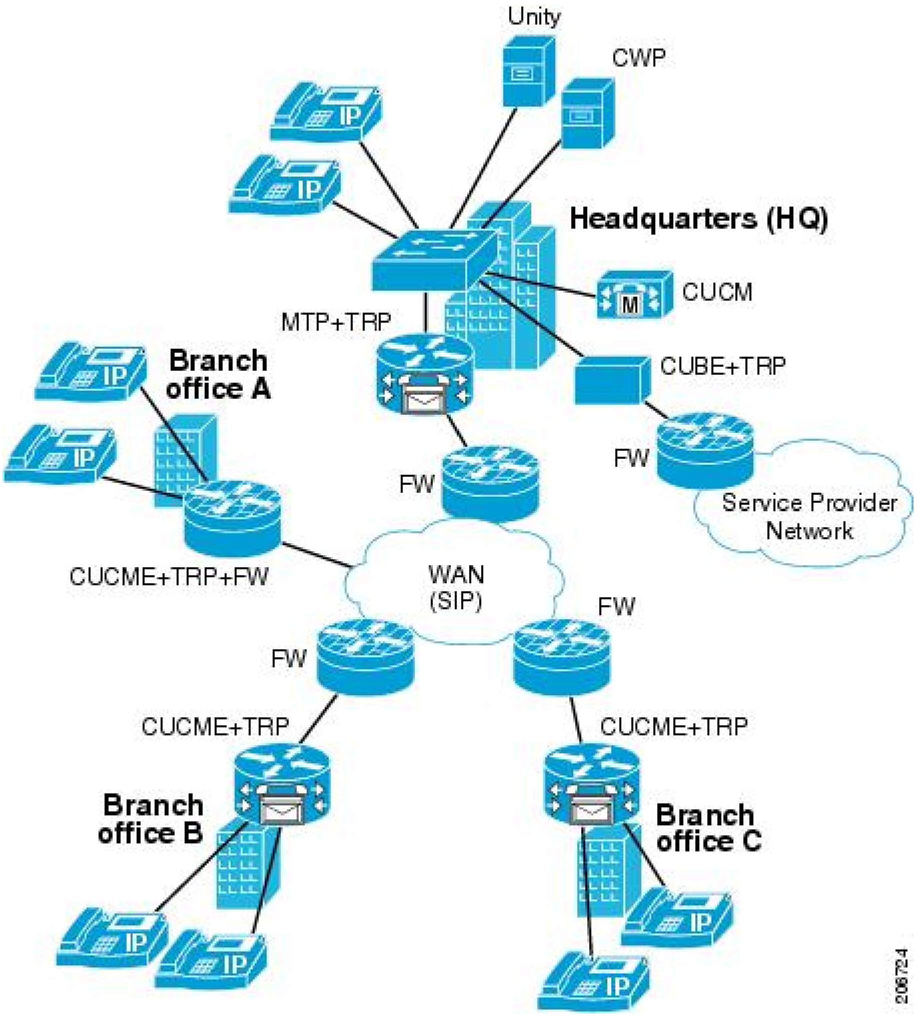
How Cisco IOS Firewall Supports TRP in a Voice Network

The following information describes the deployment scenarios supported by the Cisco IOS Firewall with TRP present in a voice network:

- For the Cisco IOS Firewall that is running on a Cisco router without TRP, STUN packets are processed as regular passthrough packets. To open a pinhole for secondary channels, the firewall must be able to recognize the STUN packets.
- For the Cisco IOS Firewall that is running on a Cisco router with TRP (Branch Office A in the figure below), the firewall will intercept and act on the STUN packets that are sent from the TRP on its WAN side. Cisco IOS firewall validates the Cisco Proprietary Cisco Flow-Data information on the STUN packet and opens the data-channel pinholes for voice traffic. The Cisco Flow-Data has information to authenticate that the message is from a valid TRP device.
- The phones do not yet support STUN. If the firewall has to open pinholes between phones, TRP should send one-sided STUN messages addressed to each phone so the firewall can see the messages and open

the pinholes. Without the support of STUN messages from TRP, the firewall would not be able to open the necessary pinholes for the phones to communicate.

Figure 8: Architecture for Cisco IOS Firewall in a TRP Network Solution



How Cisco IOS Firewall Supports Partial SIP Inspection

Cisco IOS Firewall TRP support enables Cisco IOS Firewall to process UDP STUN messages that open pinholes for secondary channels, which are necessary for implementation of TRPs in voice networks.

Previous implementations of Cisco IOS Firewall, SIP clients could negotiate with the server to dynamically open control channels on a port, which could not be supported using the access-group class map. In addition, SIP traffic was sent through the firewall without any protocol conformance checks.

To overcome these issues, Cisco IOS Firewall supports partial SIP inspection. This allows the SIP Application-level Gateway (ALG) to parse the entire SIP message, including the Session Description Protocol (SDP) part to check for protocol conformance, but does not allows SIP ALG to open pinholes for media information found in the SDP message. The STUN ALG is allowed to open the pinholes in the firewall.

Because partial SIP inspection decouples the media channel from the SIP control channel, SIP ALG can no longer depend on media channel inactivity to timeout the control sessions. Therefore, the SIP ALG implementation in this environment depends on the UDP timeout configured on the router. Because the default setting is low (30 seconds), you must set the UDP timeout value to a value slightly longer than the SIP call duration, when configuring the system.

Note In Cisco IOS Release 12.4(22)T, if you need to allow SIP control traffic, you must configure the match access-group filter. This filter allows SIP traffic to pass through the firewall without the protocol conformance check (Deep Packet Inspection).

TRP Messages

TRP uses the following message types to control how the Cisco IOS Firewall manages sessions:

- Keep-Alive messages

To keep the Cisco IOS Firewall media sessions active the TRP generates authenticated keep-alive messages which must be validated to keep the session open. The keep-alive messages are valid only for a configured length of time, which is configured on the call-control entity (CCE). The Cisco IOS Firewall must receive a new message within the configured time, otherwise it closes the pinhole. The keep-alive message has the Cryptographic Authentication Token (CAT) obtained from the CCE which must be validated by the Cisco IOS Firewall before the keep-alive message is accepted.

- Periodic Open messages

The CAT (obtained from the CCE) is valid only for the CAT-life seconds setting configured on the CCE. After that time TRP gets a new CAT and sends a new message with the new CAT. This periodic open message specifies the keys that the Cisco IOS Firewall uses to authenticate the keep-alive messages until the next new CAT is obtained. Therefore, if the Cisco IOS Firewall does not receive a new CAT with the time specified by the CAT-life seconds, the media session closes as it cannot authenticate any keep-alive messages.

- Close pinhole message

If the Cisco IOS Firewall receives a STUN message from TRP that indicates that a session should be active for 0 seconds (Seconds-Active = 0), it first validates the packet, then generates a syslog message and then allows the message to pass through the Cisco IOS Firewall so that other firewalls on the path can also see the message and close their session, finally it closes the session.

- STUN messages from a remote party

When TRP is configured on both the caller and the called side, the Cisco IOS Firewall receives 2 STUN messages for the same session. The Cisco IOS Firewall does not validate STUN messages from the remote party, instead it drops the packets and generates a syslog message.

How to Configure a Firewall to Support TRP in Voice Networks

Configuring a Policy to Allow STUN Messages

Perform this task to configure a policy to allow STUN messages.

Before You Begin

If the firewall is configured on the same device as the TRP, the STUN policy needs to be applied on the zone-pair between self and out zones.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map type inspect** [**match any**| **match all**] *class-map-name*
4. match protocol stun-ice *stun-ice-parameter-map*
5. **exit**
6. **class-map type inspect** [**match any**| **match all**] *class-map-name*
7. match access-group {access-group | name access-group-name}
8. match protocol stun-ice *stun-ice-parameter-map*
9. **exit**
10. policy-map type inspect policy-map-name
11. class type inspect class-name
12. **inspect**
13. **exit**
14. class type inspect class-name
15. **inspect**
16. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose

Step 3	**class-map type inspect** [**match any**| **match all**] *class-map-name*
	Example:
	`Router(config)# class-map type inspect stun-traffic`
		Creates an inspect type class map and enters class-map configuration mode.

Step 4	match protocol stun-ice *stun-ice-parameter-map* **Example:** `Router(config-cmap)# match protocol stun-ice cfd1`	Configures the match criteria for a class map on the basis of a specified protocol.
Step 5	**exit** **Example:** `Router(config-cmap)# exit`	Exits class-map configuration mode.

Step 6	**class-map type inspect** [**match any**| **match all**] *class-map-name*
	Example:
	`Router(config)# class-map type inspect voice-control-traffic`
		Creates an inspect type class map and enters class-map configuration mode.

Step 7	match access-group {access-group | name access-group-name}
	Example:
	`Router(config-cmap)# match access-group 101`
		Configures the match criteria for a class map based on the ACL name or number.

Step 8	match protocol stun-ice *stun-ice-parameter-map* **Example:** `Router(config-cmap)# match protocol stun-ice cfd2`	Configures the match criteria for a class map on the basis of a specified protocol.
Step 9	**exit** **Example:** `Router(config-cmap)# exit`	Exits class-map configuration mode.
Step 10	policy-map type inspect policy-map-name **Example:** `Router(config)# policy-map type inspect voice-traffic`	Creates an inspect type policy map and enters policy map configuration mode.

	Command or Action	Purpose
Step 11	class type inspect class-name **Example:** `Router(config-pmap)# class type inspect voice-control-traffic`	Specifies the traffic (class) on which an action is to be performed.
Step 12	**inspect** **Example:** `Router(config-pmap-c)# inspect`	Enables Cisco IOS stateful packet inspection.
Step 13	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits policy map class configuration mode.
Step 14	class type inspect class-name **Example:** `Router(config-pmap)# class type inspect stun-traffic`	Specifies the traffic (class) on which an action is to be performed.
Step 15	**inspect** **Example:** `Router(config-pmap-c)# inspect`	Enables Cisco IOS stateful packet inspection.

Step 16	**exit**
	Example:
	`Router(config-pmap-c)# exit`
	Example:
	`Router(config-pmap)# exit`
		Exits policy map class and policy map configuration mode.

Configuring Maps to Allow Partial SIP Inspection

Perform this task to define a parameter map that does not create or open a media channel when the parameter map is attached to the SIP class map.

Before You Begin

Because partial SIP inspection decouples the media channel from the SIP control channel, SIP ALG can no longer depend on media channel inactivity to timeout the control sessions. Therefore, the SIP ALG implementation in this environment depends on the UDP timeout configured on the router. Because the default setting is low (30 seconds), you must set the UDP timeout value to a value slightly longer than the SIP call duration, when configuring the system.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type protocol-info sip** *parameter-map-name*
4. disable open-media-channel
5. **exit**
6. **class-map type inspect** *class-map-name*
7. match protocol sip *parameter-map-name*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type protocol-info sip** *parameter-map-name* **Example:** `Router(config)# parameter-map type protocol-info sip pmap-sip`	Defines a SIP-protocol-info parameter map and enters parameter map type configuration mode.
Step 4	disable open-media-channel **Example:** `Router(config-profile)# disable open-media-channel`	Prevents the creation of a media channel when this parameter map is attached to the SIP class map.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Router(config-profile)# exit`	Exits parameter map type configuration mode.

Step 6	**class-map type inspect** *class-map-name*
	Example:
	`Router(config)# class-map type inspect cmap-sip-traffic`
		Creates an inspect type class map and enters class-map configuration mode.

Step 7	match protocol sip *parameter-map-name* **Example:** `Router(config-cmap)# match protocol sip pmap-sip`	Configures the match criteria for a class map on the basis of a specified protocol.
Step 8	**exit** **Example:** `Router(config-cmap)# exit`	Exits class-map configuration mode.

Configuring a Parameter Map for TRP Support

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **parameter-map type protocol-info stun-ice** *parameter-map-name*
4. authorization agent-id shared-secret password cat-window number

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**parameter-map type protocol-info stun-ice** *parameter-map-name* **Example:** `Router(config)# parameter-map type protocol-info stun-ice abc1`	Defines an application-specific parameter map and enters parameter map type configuration mode.
Step 4	authorization agent-id shared-secret password cat-window number **Example:** `Router(config-profile)# authorization agent-id 20 shared-secret 12345flower12345 cat-window 15`	Configures the credentials of more than one authorization agent in the same parameter map and associates the same credentials with the filter that was set up via the **match protocol stun-ice** command.

Configuration Examples for Firewall and TRP in a Voice Network

Example Cisco IOS Firewall Support of STUN Messages in Voice Network Configuration

The following example shows how to configure a Cisco IOS Firewall policy to support STUN messages:

```
parameter-map type protocol-info stun-ice abc1
 authorization agent-id 10 password letmein CAT-window 3
class-map type inspect stun-traffic
 match protocol stun-ice abc1
class-map type inspect voice-control-traffic
 match access-group 101
 match protocol udp
policy-map type inspect voice-traffic
 class type inspect voice-control-traffic
  inspect
 class type inspect stun-traffic
  inspect
access-list 101 permit ip 10.0.0.0 255.255.255.255 2.2.2.2 255.255.255.255
! Allow SIP control packets to ensure the Cisco IOS firewall does not open secondary !
channels for media.
!
access-list 101 permit tcp any any eq 5060
access-list 101 permit udp any any eq 5060
!
class-map type inspect voice-control-traffic
 match access-group 101
```

```
!
policy-map type inspect policy_test
 class type inspect voice-control-traffic
  inspect
```

Additional References

The following sections provide references related to the Cisco IOS Firewall Support for TRP feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Additional firewall commands	*Cisco IOS Security Command Reference*
Zone-based policy firewall	"Zone-Based Policy Firewall"

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	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

RFC	Title
None	

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Firewall Support for TRP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 12: Feature Information for Firewall Support for TRP

Feature Name	Releases	Feature Information
Cisco IOS Firewall Support for TRP--Phase 1	12.4(11)T	This feature enables Cisco IOS Firewall to process STUN messages. STUN messages open pinholes for secondary channels, which are necessary for implementation of TRPs in voice networks. The following commands were introduced or modified: **authorization agent-id**, **match protocol**, **parameter-map type**.
Cisco IOS Firewall Support for TRP--Phase 2	15.0(1)M	This feature enables Cisco IOS Firewall to perform partial SIP inspection and modifies some processes that were introduced in Phase 1. The following commands were introduced or modified: **parameter-map type protocol-info** , **disable open-media-channel**.

CHAPTER 8

Firewall ACL Bypass

The Firewall ACL Bypass feature allows a packet to avoid redundant access control list (ACL) checks by allowing the firewall to permit the packet on the basis of existing inspection sessions instead of dynamic ACLs. Thus, input and output dynamic ACLs searches are eliminated, improving the overall throughput performance of the base engine.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Firewall ACL Bypass

Benefits of Firewall ACL Bypass

Because input and output dynamic ACLs are no longer necessary, the need for context-based access control (CBAC) to create dynamic ACLs on the interface is eliminated. Thus, the following benefits are now available:

- Improved connections per second performance of the firewall
- Reduced run-time memory consumption of the firewall

Firewall ACL Bypass Functionality Overview

Before ACL bypassing was implemented, a packet could be subjected to as many as three redundant searches--an input ACL search, an output ACL search, and an inspection session search. Each dynamic ACL that CBAC creates corresponds to a single inspection session. Thus, a matching dynamic ACL entry for a given packet implies that a matching inspection session exists and that the packet should be permitted through the ACL. Because a matching inspection session is often found in the beginning of IP processing, the input and output dynamic ACL searches are no longer necessary and can be eliminated.

ACL bypassing subjects the packet to one search--the inspection session search--during its processing path through the router. When a packet is subjected to a single inspection session search before the ACL checks, the packet is matched against the list of session identifiers that already exist on the interface. (Session identifiers keep track of the source and destination IP addresses and ports of the packets and on which interface the packet arrived.)

Note Session identifiers are not created on interfaces for inspection sessions that are only Intrusion Detection Sessions (IDS).

How to Configure Firewall ACL Bypass

After your firewall is configured for inspection, ACL bypassing is performed by default. That is, you should configure inspection as normal.

To configure CBAC for your firewall, see the following chapter "Configuring Context-Based Access Control" in the *Cisco IOS Security Configuration Guide*

Configuration Examples for Verifying Firewall Session Information

After you have configured your firewall for inspection, you can use the **show ip inspect sessions detail** command to view session inspection information. The following examples show how eliminating dynamic ACLs changes the sample output:

Example Old showipinspect CLI Output

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
Router# show access-lists

Extended IP access list 101
    permit tcp host 192.168.101.115 eq telnet host 192.168.1.116 eq 32956 (27 matches)
    deny udp any any
    deny tcp any any
    permit ip any any
Extended IP access list 102
    permit tcp host 192.168.101.115 eq telnet host 192.168.1.116 eq 32956 (27 matches)
    deny udp any any
    deny tcp any any
    permit ip any any
```

Example New show ip inspect CLI Output

The following is sample output from the **show ip inspect session detail**command, which shows related ACL information (such as session identifiers [SID]), 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
Router# show access-list

Extended IP access list 101
    deny udp any any (20229 matches)
    deny tcp any any
    permit ip any any (6 matches)
Extended IP access list 102
    deny udp any any
    deny tcp any any
    permit ip any any (1 match)
```

Additional References

Related Documents

Related Topic	Document Title
Cisco commands	Master Command List, All Releases
Security commands	• Security Command Reference: Commands A to C • Security Command Reference: Commands D to L • Security Command Reference: Commands M to R • Security Command Reference: Commands S to Z

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Firewall ACL Bypass

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 13: Feature Information for Firewall ACL Bypass

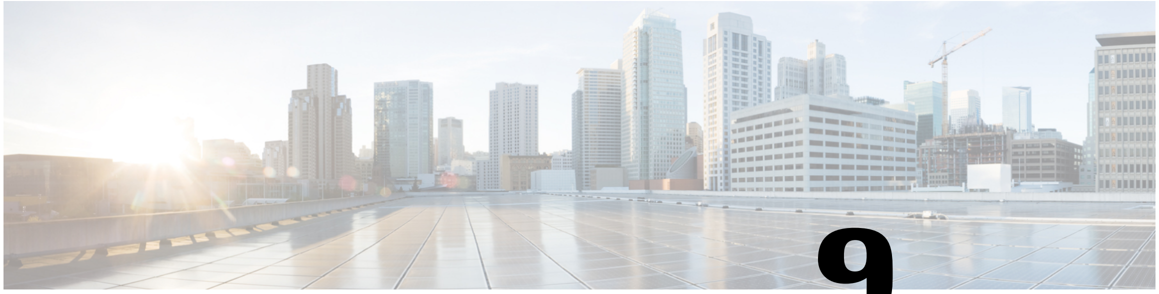
Feature Name	Releases	Feature Information
Firewall ACL Bypass	12.3(4)T	The Firewall ACL Bypass feature allows a packet to avoid redundant access control list (ACL) checks by allowing the firewall to permit the packet on the basis of existing inspection sessions instead of dynamic ACLs. Thus, input and output dynamic ACLs searches are eliminated, improving the overall throughput performance of the base engine. The following commands were introduced or modified: **show ip inspect**.

Glossary

connections per second -- Metric defined by the number of short-lived connections that can be created and deleted within 1 second by a router running CBAC. (These connections apply only to TCP connections because UDP is a connectionless protocol.)

throughput --Metric defined by the number of packets transferred from the input interface to the output interface within 1 second by a router running CBAC.

Note Refer to Internetworking Terms and Acronyms for terms not included in this glossary.



CHAPTER 9

Firewall Websense URL Filtering

The Firewall Websense URL Filtering feature enables the firewall (also known as Cisco Secure Integrated Software) to interact with the Websense URL filtering software, thereby allowing you to prevent users from accessing specified websites on the basis of a configured policy. The firewall works with the Websense server to recognize whether a particular URL should be allowed or denied (blocked).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Firewall Websense URL Filtering

Websense Server Requirement

To enable this feature, you must have at least one Websense server; however, two or more Websense servers are preferred. Although there is no limit to the number of Websense servers you may have, and you can configure as many servers as you wish, only one server will be active at any given time—the primary server. URL lookup requests are sent only to the primary server.

URL Filtering Support

Before enabling the Firewall Websense URL Filtering feature, you must ensure that there is no other URL filtering scheme configured, such as N2H2.

Restrictions for Firewall Websense URL Filtering

URL Filtering Support Restriction

This feature supports only one active URL filtering scheme at a time.

Username Restriction

This feature does not pass the username and group information to the Websense server. However, the Websense server can work for user-based policies because it has a mechanism for getting the username to correspond to an IP address.

Exclusive Domain List Restriction

Websense URL filtering does not resolve domains before it searches for an exclusive domain list. When a questionable URL is presented to the filtering server, Websense URL filtering searches only for the value that was specified in the CLI. For example, if an exclusive domain list was configured by using the **ip urlfilter exclusive-domain deny 192.168.1.1** command, a user typing http://192.168.1.1 into a browser's address field will be denied access. However, a user who is trying to access the same domain and who enters http://www.cisco.com will be allowed access because 192.168.1.1 was specified via the CLI and not www.cisco.com.

PISA URL Filtering Restrictions—Cisco IOS Release 12.2(18)ZYA

- Context-based Access Control (CBAC) is not supported.
- HTTP over ports that are used by static Network-Based Application Recognition (NBAR) protocols are not supported.
- Only HTTP filtering is supported. HTTPS and FTP filtering are not supported.
- Only Layer 3 switch virtual interfaces (SVIs), Layer 3 routed ports, and Layer 3 subinterfaces are supported.
- Only one inspection rule is supported.

- Only the Websense URL filtering server is supported. N2H2, SmartFilter, and Trend Micro filtering servers are not supported.
- The **clear ip urlfilter cache** and **show ip urlfilter cache** commands are not supported.
- Usernames are not passed on from the Programmable Intelligent Services Accelerator (PISA) to the Websense server.

Information About Firewall Websense URL Filtering

Benefits of Firewall Websense URL Filtering

The Firewall Websense URL Filtering feature provides an Internet management application that enables you to control web traffic for a given host or user on the basis of a specified security policy.

Websense is a third-party filtering software that can filter HTTP requests on the basis of the following policies: destination hostname, destination IP address, keyword, and username. The software maintains a URL database for more than 20 million sites that are organized into more than 60 categories and subcategories. This feature supports the following functionalities:

Primary and Secondary Servers

When users configure multiple Websense servers, the firewall will use only one server at a time—the primary server; all other servers are called secondary servers. When the primary server becomes unavailable for any reason, it becomes a secondary server and one of the secondary servers becomes the primary server.

The firewall marks the primary server as down when sending a request to or receiving a response from the server fails. When the primary server goes down, the firewall goes to the beginning of the configured servers list and tries to activate the first server on the list. If the first server on the list is unavailable, it will try to activate the second server on the list; the system keeps trying to activate a server until it is successful or until it reaches the end of the server list. If the system reaches the end of the server list without activating any server, the system will set a flag indicating that all servers are down, and the system will enter allow mode.

When all servers are down and the system is in allow mode, a periodic event that occurs every minute will trace through the server list, trying to bring up a server by opening a TCP connection. If a TCP connection is successfully opened, the server is considered to be up, and the system will return to operational mode.

IP Cache Table

An IP cache table contains IP addresses of web servers whose underlying URLs can be accessed by all users and hosts.

The caching algorithm involves three parameters—the maximum number of IP addresses that can be cached, an idle time, and an absolute time. The algorithm also involves two timers—idle timer and absolute timer. The idle timer is a small periodic timer (1 minute) that checks to see whether the number of cached IP addresses in the cache table exceeds 80 percent of the maximum limit. When the number of cached IP addresses exceed 80 percent, the idle timer starts removing idle entries; if the number of cached IP addresses do not exceed 80 percent, the idle timer quits and waits for the next cycle. The absolute timer is a large periodic timer (1 hour) that removes all elapsed entries. (The age of an elapsed entry is greater than the absolute time.) An elapsed entry is also removed during cache lookup.

The idle time value is fixed at 10 minutes. The absolute time value is taken from the Websense lookup response, which is often greater than 15 hours. The absolute value for a cache entry that is made of exclusive domains is 12 hours. The maximum number of cache entries is configurable.

To configure cache table parameters, use the **ip urlfilter cache** command.

Note For a device to cache pages when using the Firewall Websense URL Filtering feature, the Websense server must send the Websense cache command bit as 1. Use the **show ip urlfilter** command to display the statistics of cached entries.

Packet Buffering

Packet buffering enables you to increase the maximum number of HTTP responses that a firewall can hold. If HTTP responses arrive before a Websense server reply, the buffering scheme allows the firewall to store a maximum of 200 HTTP responses. After 200 responses have been reached, the firewall will drop further responses. Responses remain in the buffer until an allow or deny message is received from the Websense server. If the message indicates that the URL is allowed, the firewall will release HTTP responses in the buffer to the browser of the end user. If the message indicates that the URL is blocked, the firewall discards HTTP responses in the buffer and closes the connection to both ends. Packet buffering prevents numerous HTTP responses from overwhelming your system.

To configure the maximum number of HTTP responses for the firewall, use the **ip urlfilter max-resp-pak** command.

Exclusive Domains

Exclusive domains provides a configurable list of domain names so that the firewall does not have to send a lookup request to the Websense server for the HTTP traffic that is destined for one of the domains in the exclusive list. Thus, the Websense server does not need to handle lookup requests for HTTP traffic that is destined for a host that has already been marked as “allowed.”

Flexibility when entering domain names is also provided; that is, you can enter the complete domain name or a partial domain name. If the user adds a complete domain name, such as “www.cisco.com,” to the exclusive domain list, all HTTP traffic whose URLs are destined for this domain (such as www.cisco.com/news and www.cisco.com/index) will be excluded from the Websense URL filtering policies, and based on the configuration, the URLs will be permitted or blocked (denied).

If the user adds a partial domain name such as “.cisco.com” to the exclusive domain list, all URLs whose domain names end with this partial domain name (such as www.cisco.com/products and www.cisco.com/eng) will be excluded from the Websense URL filtering policies, and based on the configuration, the URLs will be permitted or blocked (denied).

To configure an exclusive domain list, use the **ip urlfilter exclusive-domain** command.

Allow Mode

A system enters allow mode when connections to all Websense servers are down. The system will return to normal mode when a connection to at least one Websense server is up. Allow mode directs your system to forward or drop all packets on the basis of the configurable allow mode setting. By default, allow mode is off, so all HTTP requests are forbidden if all Websense servers are down.

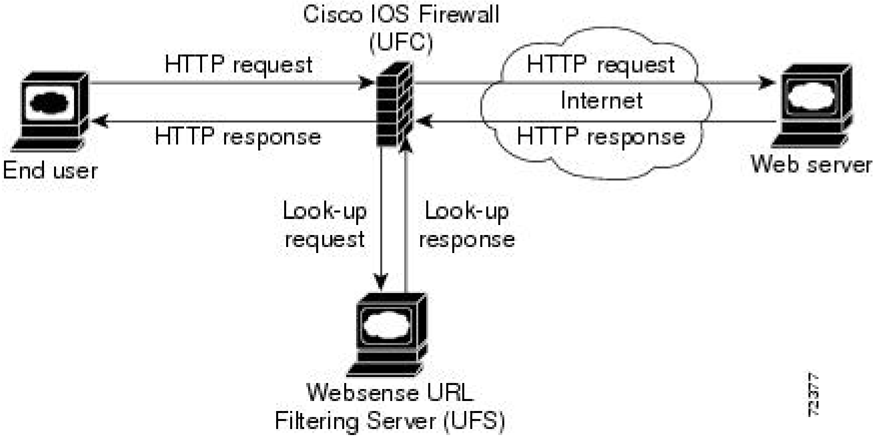
To configure allow mode for the system, use the **ip urlfilter allowmode** command.

Feature Design of Firewall Websense URL Filtering

Note The Firewall Websense URL Filtering feature assumes that the Websense server will be part of a protected network and that requests from the firewall will not travel over any unprotected network to reach the Websense server.

The figure below and the corresponding steps explain a sample URL filtering network topology.

Figure 9: Firewall Websense URL Filtering Sample Topology



1 The end user browses a page on the web server, and the browser sends an HTTP request.

2 After the firewall receives this request, it forwards the request to the web server while simultaneously extracting the URL and sending a lookup request to the Websense server.

3 After the Websense server receives the lookup request, it checks its database to see whether it should permit or deny the URL; it returns a permit or deny status via a lookup response to the firewall.

4 After the firewall receives the lookup response, it performs one of the following actions:

- If the lookup response permits the URL, the firewall sends the HTTP response to the end user.
- If the lookup response denies the URL, the Websense server redirects the user to its own internal web server, which displays a message that describes the category under which the URL is blocked; thereafter, the connection is reset on both ends.

Websense Server Features Supported on a Firewall

The firewall supports all filtering and user authentication methods that are supported by the Websense server.

The following filtering methods are supported:

- Category-based filtering that is applied on the basis of specific categories.
- Customized filtering that allows a user to apply a policy for customized URLs.

- Global filtering that is applied to all IP addresses, groups, and users.
- Keyword-based filtering that is applied on the basis of specific keywords (for example, a user can configure a policy to deny all URLs with the keyword “spam”).
- User- or group-based filtering that is applied to a specific user or group.

The Websense server feature supports the NT LAN Manager (NTLM) and Lightweight Directory Access Protocol (LDAP) user authentication methods. The Websense server uses these methods to authenticate a user when the firewall does not pass the authenticated username along with the lookup request.

When the username is not passed along with the lookup request, the Websense server retrieves the username using one of the following methods:

- Manual authentication—The Websense server redirects the user to its own internal web server, which displays a challenge or response for the username and password. (This process is similar to when a user is blocked, but in this process, an authentication message is displayed instead of a blocked message.) Thereafter, the Websense server checks the NTLM or LDAP directory service to see if the username and password match. If there is a match, the Websense server associates the username with the source IP address and creates policies for this username.
- Transparent ID (XID) authentication—The Websense server has an agent that automatically associates users, when they log in to a Windows network, to their IP addresses. Unlike manual authentication, this method does not require an additional login by the user. However, this method can be used only for Windows.

Note Although the Websense server also supports user authentication via TACACS or RADIUS, this feature currently does not support these protocols for user authentication.

How to Configure Firewall Websense URL Filtering

Configuring Firewall Websense URL Filtering

Before You Begin

Before enabling the Firewall Websense URL Filtering feature, ensure that no other URL filtering scheme is configured, such as N2H2. If you try to enter a new filtering scheme when one already exists, the new scheme will be ignored, and the system will display an error message that says, “different URL filtering scheme cannot coexist.”

Note Enabling HTTP inspection (by using the **ip inspect name** command) 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* keyword-argument pair with the **ip inspect name** command and configure a standard access list to allow any traffic. Configuring URL filtering without enabling the **java-list** *access-list* keyword-argument pair will severely impact performance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-name* **http** [**java-list** *access-list*] [**urlfilter**] [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]
4. **ip inspect** *inspection-name* {**in** | **out**}
5. **ip urlfilter server vendor** {**websense** | **n2h2**} *ip-address* [**port** *port-number*] [**timeout** *seconds*] [**retransmit** *number*]
6. **ip urlfilter alert**
7. **ip urlfilter audit-trail**
8. **ip urlfilter urlf-server-log**
9. **ip urlfilter exclusive-domain** {**permit** | **deny**} *domain-name*
10. **ip urlfilter cache** *number*
11. **ip urlfilter allowmode** [**on** | **off**]
12. **ip urlfilter max-resp-pak** *number*
13. **ip urlfilter max-request** *number*
14. **ip urlfilter truncate** {**script-parameters** | **hostname**}
15. **ip urlfilter mode** {**per-session** | **per-uri** | **per-uri-proxy-only**}
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect name** *inspection-name* **http** [**java-list** *access-list*] [**urlfilter**] [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*] **Example:** `Device(config)# ip inspect name fw-urlf http java-list 51 urlfilter timeout 30`	Enables HTTP inspection. • The **urlfilter** keyword associates URL filtering with HTTP inspection. **Note** You can configure two or more inspections on a device, but URL filtering will work only with inspections in which the **urlfilter** keyword is enabled.

	Command or Action	Purpose
		Note Enabling HTTP inspection with or without any options 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* keyword-argument pair. Configuring URL filtering without enabling the **java-list** *access-list* keyword-argument pair will severely impact performance.

Step 4	**ip inspect** *inspection-name* {**in** | **out**}
	Example:
	`Device(config)# ip inspect fw-urlf in`
		Applies a set of inspection rules to an interface.

Step 5	**ip urlfilter server vendor** {**websense** | **n2h2**} *ip-address* [**port** *port-number*] [**timeout** *seconds*] [**retransmit** *number*] **Example:** `Device(config)# ip urlfilter server vendor websense 10.201.6.202`	Configures a Websense server to interact with the firewall to filter HTTP requests on the basis of a specified policy.
Step 6	**ip urlfilter alert** **Example:** `Device(config)# ip urlfilter alert`	(Optional) Enables the system alert, which displays system messages such as a server entering allow mode or going down.
Step 7	**ip urlfilter audit-trail** **Example:** `Device(config)# ip urlfilter audit-trail`	(Optional) Enables the logging of messages into the syslog server of a device.
Step 8	**ip urlfilter urlf-server-log** **Example:** `Device(config)# ip urlfilter urlf-server-log`	(Optional) Enables the logging of system messages on the URL filtering server (the Websense server).
Step 9	**ip urlfilter exclusive-domain** {**permit** | **deny**} *domain-name* **Example:** `Device(config)# ip urlfilter exclusive-domain permit www.cisco.com`	(Optional) Adds a domain name to or from an exclusive domain list so that the firewall does not have to send lookup requests to the Websense server.

	Command or Action	Purpose
Step 10	**ip urlfilter cache** *number* **Example:** `Device(config)# ip urlfilter cache 4500`	(Optional) Configures cache table parameters.
Step 11	**ip urlfilter allowmode** [**on** | **off**] **Example:** `Device(config)# ip urlfilter allowmode on`	(Optional) Enables the default mode of filtering systems.
Step 12	**ip urlfilter max-resp-pak** *number* **Example:** `Device(config)# ip urlfilter max-resp-pak 150`	(Optional) Configures the maximum number of HTTP responses that the firewall can keep in its packet buffer.
Step 13	**ip urlfilter max-request** *number* **Example:** `Device(config)# ip urlfilter max-request 500`	(Optional) Sets the maximum number of outstanding requests that can exist at any given time. • If the maximum number of requests is reached, all subsequent URLs are dropped.

Step 14	**ip urlfilter truncate** {**script-parameters** | **hostname**}
	Example:
	`Device(config)# ip urlfilter truncate hostname`
		(Optional) Allows the URL filter to truncate long URLs to the server.

Step 15	**ip urlfilter mode** {**per-session** | **per-uri** | **per-uri-proxy-only**}
	Example:
	`Device(config)# ip urlfilter mode per-uri`
		(Optional) Configures a URL filtering mode.
		Note This command is available only on the Catalyst 6500 with PISA.

Step 16	**end**
	Example:
	`Device(config)# end`
		Exits global configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

This feature introduces the following alert messages:

- "%URLF-3-SERVER_DOWN: Connection to the URL filter server 10.92.0.9 is down"
  This LOG_ERR-type message is displayed when a configured URL filter server (UFS) goes down. When the UFS goes down, the firewall marks the configured server as secondary and tries to bring up one of the other secondary servers and marks that server as the primary server. If no other server is configured, the firewall enters allow mode and displays the "URLF-3-ALLOW_MODE" message.

- "%URLF-3-ALLOW_MODE: Connection to all URL filter servers are down and ALLOW MODE is OFF"
  This LOG_ERR-type message is displayed when all UFSs are down and the system enters allow mode.

Note Whenever the system goes into allow mode (all filter servers are down), a periodic keepalive timer is triggered, which tries to bring up a server by opening a TCP connection.

- "%URLF-5-SERVER_UP: Connection to the URL filter server 10.92.0.9 is made; the system is returning from ALLOW MODE"
  This LOG_NOTICE-type message is displayed when UFSs are detected as being up and the system is returning from allow mode.

- "%URLF-4-URL_TOO_LONG: URL too long (more than 3072 bytes), possibly a fake packet?"
  This LOG_WARNING-type message is displayed when the URL in a lookup request is too long; any URL longer than 3K is dropped.

- "%URLF-4-MAX_REQ: The number of pending requests exceeds the maximum limit <1000>"
  This LOG_WARNING-type message is displayed when the number of pending requests in the system exceeds the maximum limit and all further requests are dropped.

To display these alert messages, use the **ip urlfilter alert** command. This feature introduces the following syslog messages:

- "%URLF-6-SITE_ALLOWED: Client 10.0.0.2:12543 accessed server 10.76.82.21:8080"
  This message is logged for each request whose destination IP address is found in the cache. It includes the source IP address, source port number, destination IP address, and destination port number. The URL is not logged because the IP address of the request is found in the cache, so parsing the request and extracting the URL is a waste of time.

- "%URLF-4-SITE-BLOCKED: Access denied for the site 'www.sports.com'; client 10.54.192.6:34557 server 172.24.50.12:80"
  This message is logged when a request finds a match against one of the blocked domains in the exclusive-domain list or the corresponding entry in the IP cache.

- "%URLF-6-URL_ALLOWED: Access allowed for URL http://www.websense.com/; client 10.54.192.6:54123 server 192.168.0.1:80"
  This message is logged for each URL request that is allowed by a UFS. It includes the allowed URL, source IP address, source port number, destination IP address, and destination port number. Longer URLs will be truncated to 300 bytes and then logged.

- "%URLF-6-URL_BLOCKED: Access denied URL http://www.google.com; client 10.45.192.6:54678 server 192.168.0.1:80"

> This message is logged for each URL request that is blocked by a UFS. It includes the blocked URL, source IP address, source port number, destination IP address, and destination port number. Longer URLs are truncated to 300 bytes and then logged.

To display these syslog messages, use the **ip urlfilter audit-trail** command.

Verifying and Monitoring Firewall Websense URL Filtering

To verify that the Firewall Websense URL Filtering feature is working, perform any of the following optional steps. You can use these commands in any order.

Command or Action	Purpose
enable `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
show ip urlfilter cache `Device# show ip urlfilter cache`	Displays destination IP addresses that are cached in the cache table.
show ip urlfilter config `Device# show ip urlfilter config`	Displays the size of the cache, the maximum number of outstanding requests, the allow mode state, and the list of configured Websense servers.
show ip urlfilter statistics `Device# show ip urlfilter statistics`	Displays information such as the number of requests that are sent to the Websense server, the number of responses received from the Websense server, the number of pending requests in the system, the number of failed requests, and the number of blocked URLs.

debug ip urlfilter {**function-trace** | **detailed** | **events**}
`Device# debug ip urlfilter detailed`
	Enables the debugging of the URL filter subsystems information.

clear ip urlfilter cache {*ip-address* | **all**} `Device# clear ip urlfilter cache all`	Clears the cache table.

Configuration Examples for Firewall Websense URL Filtering

Example: Configuring Firewall Websense URL Filtering

```
hostname fw9-7200b
!
logging buffered 64000 debugging
enable secret 5 $1$qMOf$umPb75mb3sV27JpNbW//7.
!
clock timezone PST -8
clock summer-time PDT recurring
ip subnet-zero
ip cef
no ip domain lookup
!
ip inspect name test http urlfilter
ip urlfilter cache 5
ip urlfilter exclusive-domain permit .example-1.com
ip urlfilter exclusive-domain deny .example-2.com
ip urlfilter exclusive-domain permit www.example.com
ip urlfilter audit-trail
ip urlfilter alert
ip urlfilter server vendor websense 192.168.3.1
ip audit notify log
ip audit po max-events 100
ip port-map http port 8080
!
no voice hpi capture buffer
no voice hpi capture destination
!
mta receive maximum-recipients 0
!
interface FastEthernet 0/0
 ip address 192.168.3.254 255.255.255.0
 ip access-group 101 out
 ip nat inside
 ip inspect test in
 no ip route-cache
 no ip mroute-cache
!
interface Ethernet 1/0
 ip address 10.6.9.7 255.255.0.0
 ip nat outside
 no ip route-cache
 no ip mroute-cache
 duplex half
!
interface Ethernet 1/1
 no ip address
 no ip mroute-cache
 shutdown
 duplex half
!
interface Ethernet 1/2
 no ip address
 no ip mroute-cache
 shutdown
 duplex half
!
interface Ethernet 1/3
 no ip address
 no ip mroute-cache
 shutdown
 duplex half
!
interface Serial 2/0
```

```
 no ip address
 no ip mroute-cache
 shutdown
 dsu bandwidth 44210
 framing c-bit
 cablelength 10
 serial restart-delay 0
 fair-queue
!
ip nat pool devtest 10.6.243.21 10.6.243.220 netmask 255.255.0.0
ip nat inside source list 1 pool devtest
ip nat inside source static 192.168.3.1 10.6.243.1
ip nat inside source static 192.168.3.2 10.6.243.2
ip nat inside source static 192.168.3.3 10.6.243.3
ip classless
ip route 192.168.0.30 255.255.255.255 10.6.0.1
no ip http server
no ip http secure-server
!
ip pim bidir-enable
!
!
access-list 101 deny   tcp any any
access-list 101 deny   udp any any
access-list 101 permit ip any any
access-list 102 deny   tcp any any
access-list 102 deny   udp any any
access-list 102 permit ip any any
dialer-list 1 protocol ip permit
dialer-list 1 protocol ipx permit
!
!
call rsvp-sync
!
!
mgcp profile default
!
dial-peer cor custom
!!
!
gatekeeper
 shutdown
!
!
line con 0
 exec-timeout 0 0
 stopbits 1
line aux 0
 stopbits 1
line vty 0 4
 password letmein
 login
!
exception core-file example/exampledump/fw9-7200b.core
exception dump 192.168.0.1
no scheduler max-task-time
!
end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases

Related Topic	Document Title
Firewall commands	- Security Command Reference: Commands A to C - Security Command Reference: Commands D to L - Security Command Reference: Commands M to R - Security Command Reference: Commands S to Z
N2H2 URL filtering	"Firewall N2H2 Support" module

Standards and RFCs

Standard/RFC	Title
RFC 1945	Hypertext Transfer Protocol—HTTP/1.0
RFC 2616	Hypertext Transfer Protocol—HTTP/1.1

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Firewall Websense URL Filtering

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for Firewall Websense URL Filtering

Feature Name	Releases	Feature Information
Firewall Websense URL Filtering	12.2(11)YU 12.2(15)T 12.2(18)ZYA	The Firewall Websense URL Filtering feature enables the firewall to interact with the Websense URL filtering software, thereby allowing you to prevent users from accessing specified websites on the basis of some policy. In Cisco IOS Release 12.2(18)ZYA, support was added on the Catalyst 6500 series of switches equipped with the PISA. The following commands were introduced or modified: **clear ip urlfilter cache**, **debug ip urlfilter**, **ip inspect name**, **ip urlfilter alert**, **ip urlfilter allowmode**, **ip urlfilter audit-trail**, **ip urlfilter cache**, **ip urlfilter exclusive-domain**, **ip urlfilter max-request**, **ip urlfilter max-resp-pak**, **ip urlfilter mode**, **ip urlfilter server vendor, ip urlfilter urlf-server-log, show ip urlfilter cache**, **show ip urlfilter config**, **show ip urlfilter statistics**.

Glossary

UFC—URL filter client. UFC is a separate process that accepts URLs from CSIS, forwards the URL to the Websense server, and processes replies from the vendor server (Websense or N2H2).

UFS—URL filter server. UFS is a generic name given to the vendor server (Websense or N2H2), which processes URLs and decides whether to allow or deny web traffic on the basis of a given policy.

CHAPTER 10

HTTP Inspection Engine

The HTTP Inspection Engine feature allows users to configure their Cisco IOS Firewall to detect and prohibit HTTP connections--such as tunneling over port 80, unauthorized request methods, and non-HTTP compliant file transfers--that are not authorized within the scope of the security policy configuration. Tunneling unauthorized protocols through port 80 and over HTTP exposes a network to significant security risks.

The Cisco IOS Firewall can now be configured with a security policy that adheres to the following tasks:

- Allowing specific traffic targeted for port 80 to traverse the firewall. The traffic is inspected for protocol conformance and for the types of HTTP commands that are allowed or disallowed.
- Denying specific traffic targeted for port 80 that does not comply to HTTP traffic standards. The firewall is enabled to drop the packet, reset the connection, and send a syslog message, as appropriate.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for HTTP Inspection Engine

The Cisco 831 router with 48M RAM does not have enough memory to support this feature.

Information About HTTP Inspection Engine

Before configuring an application firewall to detect and police specific traffic targeted for port 80, you should understand the following concepts:

What Is a Security Policy

The application firewall uses a security policy, which consists of a collection of static signatures, to detect security violations. A static signature is a collection of parameters that specify protocol conditions that must be met before an action is taken. (For example, a signature may specify that an HTTP data stream containing the POST method must reset the connection.) These protocol conditions and reactions are defined by the end user via the command-line interface (CLI) to form a security policy.

Cisco IOS HTTP Application Policy Overview

HTTP uses port 80 to transport Internet web services, which are commonly used on the network and rarely challenged with regards to their legitimacy and conformance to standards. Because port 80 traffic is typically allowed through the network without being challenged, many application developers are leveraging HTTP traffic as an alternative transport protocol in which to enable their application to travel through or even bypass the firewall.

Most firewalls provide only packet filtering capabilities that simply permit or deny port 80 traffic without inspecting the data stream; the Cisco IOS application firewall for HTTP performs packet inspection as follows:

- Detects HTTP connections that are not authorized within the scope of the security policy configuration.
- Detects users who are tunneling applications through port 80.

If the packet is not in compliance with the HTTP protocol, it will be dropped, the connection will be reset, and a syslog message will be generated, as appropriate.

How to Define and Apply an HTTP Application Policy to a Firewall for Inspection

Defining an HTTP Application Policy

Use this task to create an HTTP application firewall policy.

Note Although application firewall policies are defined in global configuration mode, only one global policy for a given protocol is allowed per interface.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. appfw policy-name policy-name
4. **application** *protocol*
5. strict-http action {reset | allow} [alarm]
6. content-length {min bytes max bytes | min bytes | max bytes} action {reset | allow} [alarm]
7. content-type-verification [match-req-resp] action {reset | allow} [alarm]
8. max-header-length {request bytes response bytes} action {reset | allow} [alarm]
9. max-uri-length bytes action {reset | allow} [alarm]
10. request method {rfc rfc-method | extension extension-method} action {reset | allow} [alarm]
11. port-misuse {p2p | tunneling | im | default} action {reset | allow} [alarm]
12. transfer-encoding type {chunked | compress | deflate | gzip | identity | default} action {reset | allow} [alarm]
13. **timeout** *seconds*
14. audit-trail {on | off}
15. **exit**
16. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	appfw policy-name policy-name **Example:** `Router(config)# appfw policy-name mypolicy`	Defines an application firewall policy and puts the router in application firewall policy configuration mode.

	Command or Action	Purpose
Step 4	**application** *protocol* **Example:** `Router(cfg-appfw-policy)# application http`	Allows you to configure inspection parameters for a given protocol. Currently, only HTTP traffic can be inspected. • *protocol* --Specify the **http** keyword. This command puts you in appfw-policy-*protocol*configuration mode, where "*protocol*" is dependent upon the specified protocol. Because only HTTP can be specified, the configuration mode is appfw-policy-http.
Step 5	strict-http action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# strict-http action allow alarm`	(Optional) Allows HTTP messages to pass through the firewall or resets the TCP connection when HTTP noncompliant traffic is detected.
Step 6	content-length {min bytes max bytes | min bytes | max bytes} action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# content-length max 1 action allow alarm`	(Optional) Permits or denies HTTP traffic through the firewall on the basis of message size. • **min** | **max** *bytes*--Minimum or maximum content length, in bytes, allowed per message. Number of bytes range: 0 to 65535.
Step 7	content-type-verification [match-req-resp] action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# content-type-verification match-req-resp action allow alarm`	(Optional) Permits or denies HTTP traffic through the firewall on the basis of content message type.
Step 8	max-header-length {request bytes response bytes} action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# max-header-length request 1 response 1 action allow alarm`	(Optional) Permits or denies HTTP traffic on the basis of the message header length. • *bytes* --Number of bytes ranging from 0 to 65535.
Step 9	max-uri-length bytes action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# max-uri-length 1 action allow alarm`	(Optional) Permits or denies HTTP traffic on the basis of the URI length in the request message.
Step 10	request method {rfc rfc-method | extension extension-method} action {reset | allow} [alarm]	(Optional) Permits or denies HTTP traffic according to either the request methods or the extension methods.

	Command or Action	Purpose
	Example: `Router(cfg-appfw-policy-http)# request-method rfc default action allow alarm`	- **rfc** --Specifies that the supported methods of RFC 2616, *Hypertext Transfer Protocol--HTTP/1.1* , are to be used for traffic inspection. - *rfc-method* --Any one of the following RFC 2616 methods can be specified: **connect**, **default, delete**, **get**, **head**, **options**, **post**, **put**, **trace**. - **extension** --Specifies that the extension methods are to be used for traffic inspection. - *extension-method* --Any one of the following extension methods can be specified: **copy**, **default, edit**, **getattribute**, **getproperties**, **index**, **lock**, **mkdir**, **move**, **revadd**, **revlabel**, **revlog**, **save**, **setattribute**, **startrev**, **stoprev**, **unedit**, **unlock**.
Step 11	port-misuse {p2p | tunneling | im | default} action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# port-misuse default action allow alarm`	(Optional) Permits or denies HTTP traffic through the firewall on the basis of specified applications in the HTTP message. - **p2p** --Peer-to-peer protocol applications subject to inspection: Kazaa and Gnutella. - **tunneling** --Tunneling applications subject to inspection: HTTPPort/HTTPHost, GNU Httptunnel, GotoMyPC, Firethru, Http-tunnel.com Client - **im** --Instant messaging protocol applications subject to inspection: Yahoo Messenger. - **default** --All applications are subject to inspection.
Step 12	transfer-encoding type {chunked | compress | deflate | gzip | identity | default} action {reset | allow} [alarm] **Example:** `Router(cfg-appfw-policy-http)# transfer-encoding type default action allow alarm` **Example:**	(Optional) Permits or denies HTTP traffic according to the specified transfer-encoding of the message. - **chunked** --Encoding format (specified in RFC 2616, *Hypertext Transfer Protocol--HTTP/1*) in which the body of the message is transferred in a series of chunks; each chunk contains its own size indicator. - **compress** --Encoding format produced by the UNIX "compress" utility. - **deflate** --"ZLIB" format defined in RFC 1950, *ZLIB Compressed Data Format Specification version 3.3* , combined with the "deflate" compression mechanism described in RFC 1951, *DEFLATE Compressed Data Format Specification version 1.3* . - **gzip** --Encoding format produced by the "gzip" (GNU zip) program. - **identity** --Default encoding, which indicates that no encoding has been performed. - **default** --All of the transfer encoding types.

	Command or Action	Purpose
Step 13	**timeout** *seconds* **Example:** `Router(cfg-appfw-policy-http)# timeout 60`	(Optional) Overrides the global TCP idle timeout value for HTTP traffic. **Note** If this command is not issued, the default value specified via the **ip inspect tcp idle-time**command will be used.

Step 14	audit-trail {on | off}
	Example:
	`Router(cfg-appfw-policy-http)# audit-trail on`
		(Optional) Turns audit trail messages on or off.
		Note If this command is not issued, the default value specified via the **ip inspect audit-trail**command will be used.

Step 15	**exit** **Example:** `Router(cfg-appfw-policy-http)# exit`	Exits cfg-appfw-policy-http configuration mode.
Step 16	**exit** **Example:** `Router(cfg-appfw-policy)# exit`	Exits cfg-appfw-policy configuration mode.

What to Do Next

After you have successfully defined an application policy for HTTP traffic inspection, you must apply the policy to an inspection rule. Thereafter, the inspection rule must be applied to an interface. For information on completing this task, see the section “Applying an HTTP Application Policy to a Firewall for Inspection.”

Applying an HTTP Application Policy to a Firewall for Inspection

Use this task to apply an HTTP application policy to an inspection rule, followed by applying the inspection rule to an interface.

Note An application policy can coexist with other inspection protocols (for example, an HTTP policy and an FTP policy can coexist).

Before You Begin

You must have already defined an application policy (as shown in the section “Defining an HTTP Application Policy”).

or

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-name* **appfw** *policy-name*
4. **ip inspect name** *inspection-name* **http** [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*]
5. **interface** *type number*
6. **ip inspect** *inspection-name* **in** | **out**}
7. **exit**
8. **exit**
9. show appfw configuration [name]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect name** *inspection-name* **appfw** *policy-name* **Example:** `Router(config)# ip inspect name firewall appfw mypolicy`	Defines a set of inspection rules for the application policy. • *policy-name* --Must match the policy name specified via the **appfw policy-name** command.
Step 4	**ip inspect name** *inspection-name* **http** [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout** *seconds*] **Example:** `Router(config)# ip inspect name firewall http`	Defines a set of inspection rules that is to be applied to all HTTP traffic. • The *inspection-name* argument must match the *inspection-name* argument specified in Step 3.
Step 5	**interface** *type number* **Example:** `Router#(config)# interface FastEthernet0/0`	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose

Step 6	**ip inspect** *inspection-name* **in** | **out**}
	Example:
	`Router#(config-if)# ip inspect firewall in`
		Applies the inspection rules (defined in Step 3 and Step 4) to all traffic entering the specified interface.
		• The *inspection-name* argument must match the inspection name defined via the **ip inspect name** command.

Step 7	**exit** **Example:** `Router#(config-if)# exit`	`Exits interface configuration mode.`
Step 8	**exit** **Example:** `Router(config)# exit`	`Exits global configuration mode.`

Step 9	show appfw configuration [name]
	Example:
	`Router# show appfw configuration`
	Example:
	`or`
	Example:
	show ip inspect {**name** *inspection-name* | **config** | **interfaces** | **session** [**detail**] | **statistics** | **all**}
	Example:
	`Router# show ip inspect config`
		`(Optional) Displays application firewall policy configuration information.`
		`(Optional) Displays firewall-related configuration information.`

Troubleshooting Tips

To help troubleshoot the application firewall configuration, issue the following application-firewall specific debug command: **debug appfw**{**application** *protocol* | **function-trace** | **object-creation** | **object-deletion** | **events** | **timers** | **detailed**}.

The following sample configuration shows how to configure an HTTP policy with application firewall debugging enabled:

```
Router(config)# appfw policy-name myPolicyAPPFW  FUNC:appfw_policy_find
APPFW  FUNC:appfw_policy_find -- Policy myPolicy is not found
APPFW  FUNC:appfw_policy_alloc
APPFW  FUNC:appfw_policy_alloc -- policy_alloc 0x65727278
APPFW  FUNC:appfw_policy_alloc -- Policy 0x65727278 is set to valid
APPFW  FUNC:appfw_policy_alloc -- Policy myPolicy has been created
APPFW  FUNC:appfw_policy_command -- memlock policy 0x65727278

! Debugging sample for application (HTTP) creation

Router(cfg-appfw-policy)# application httpAPPFW  FUNC:appfw_http_command
APPFW  FUNC:appfw_http_appl_find
APPFW  FUNC:appfw_http_appl_find -- Application not found
APPFW  FUNC:appfw_http_appl_alloc
APPFW  FUNC:appfw_http_appl_alloc -- appl_http 0x64D7A25C
APPFW  FUNC:appfw_http_appl_alloc -- Application HTTP parser structure 64D7A25C created
! Debugging sample for HTTP-specific application inspection
Router(cfg-appfw-policy-http)#
Router(cfg-appfw-policy-http)# strict-http action reset alarm
APPFW  FUNC:appfw_http_subcommand
APPFW  FUNC:appfw_http_subcommand -- strict-http cmd turned on
Router# debug appfw detailed
APPFW Detailed Debug debugging is on
fw7-7206a#debug appfw object-creation
APPFW Object Creations debugging is on
fw7-7206a#debug appfw object-deletion
APPFW Object Deletions debugging is on
```

Configuration Examples for Setting Up an HTTP Inspection Engine

Example Setting Up and Verifying an HTTP Inspection Engine

The following example show 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 put 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 put 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
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Firewall commands: complete command syntax, command mode, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*

Standards

Standards	Title
No new or modified standards are supported by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported 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 2616	Hypertext Transfer Protocol -- HTTP/1.1

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Setting Up an HTTP Inspection Engine

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15: Feature Information for Setting Up an HTTP Inspection Engine

Feature Name	Releases	Feature Information
Setting Up an HTTP Inspection Engine	12.3(14)T	The HTTP Inspection Engine feature allows users to configure their Cisco IOS Firewall to detect and prohibit HTTP connections--such as tunneling over port 80, unauthorized request methods, and non-HTTP compliant file transfers--that are not authorized within the scope of the security policy configuration. Tunneling unauthorized protocols through port 80 and over HTTP exposes a network to significant security risks. The following commands were introduced or modified: **appfw policy-name**, **application**, **audit-trail**, **content-length**, **content-type-verification**, **debug appfw**, **ip inspect name**, **max-header-length**, **max-uri-length**, **port-misuse**, **request-method**, **show appfw**, **strict-http**, **timeout**, **transfer-encoding type**.

CHAPTER 11

Inspection of Router-Generated Traffic

The Inspection of Router-Generated Traffic feature allows Context-Based Access Control (CBAC) to inspect traffic that is originated by or destined to the router on which CBAC is configured. Previously, inspection of TCP, UDP, and H.323 connections initiated by or destined to the router were allowed.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Inspection of Router-Generated Traffic

- Configure CBAC.
- Configure Cisco Call Manager Express (CCME) or H.323 Gateway to configure the inspection of H.323 connections to and from the router.

Restrictions for Inspection of Router-Generated Traffic

- Inspection of router-generated traffic is supported only on the following protocols: H.323, TCP, and UDP.
- Context-Based Access Control (CBAC) firewalls do not inspect IPv6 router-generated traffic.
- CBAC firewalls support only Version 2 of the H.323 protocol. If Cisco Unified Communications Manager Express or the H.323 gateway has enabled the inspection of H.323 router traffic, configure the following commands to support only Version 2 features:

```
voice service voip
h323
session transport tcp calls-per-connection 1
h245 tunnel disable
h245 caps mode restricted
h225 timeout tcp call-idle value 0
```

Information About Inspection of Router-Generated Traffic

CBAC

CBAC is a Cisco IOS Firewall set feature that provides network protection by using the following functions:

Traffic Filtering

CBAC filters TCP and UDP packets based on application-layer protocol session information. You can configure CBAC to permit specified TCP and UDP traffic through a firewall only when the connection is initiated from within the network you want to protect. CBAC can inspect traffic for sessions that originate from either side of the firewall, and CBAC can be used for intranet, extranet, and Internet perimeters of your network.

Traffic Inspection

CBAC inspects traffic that travels through the firewall to discover and manage state information for TCP and 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.

Alerts and Audit Trails

CBAC generates real-time alerts and audit trails. Enhanced audit trail features use SYSLOG to track all network transactions; it records time stamps, the source host, the destination host, the ports used, and the total number of transmitted bytes, for advanced, session-based reporting. Real-time alerts send SYSLOG error messages to central management consoles upon detecting suspicious activity.

Using CBAC inspection rules, you can configure alerts and audit trail information on a per-application protocol basis. For example, if you want to generate audit trail information for HTTP traffic, you can specify that in the CBAC rule covering HTTP inspection.

Intrusion Detection

CBAC provides a limited amount of intrusion detection to protect against specific Simple Mail Transfer Protocol (SMTP) attacks. With intrusion detection, SYSLOG messages are reviewed and monitored for specific “attack signatures.” Certain types of network attacks have specific characteristics, or signatures. When CBAC detects an attack, it resets the offending connections and sends SYSLOG information to the SYSLOG server.

Inspection of Router-Generated Traffic Overview

Inspection of Router-Generated Traffic enhances CBAC’s functionality to inspect TCP, UDP, and H.323 connections that have a router or firewall as one of the connection endpoints. This enables CBAC to open pinholes for TCP, UDP, and H.323 control channel connections to and from the router, and to open pinholes for data and media channels negotiated over the H.323 control channels.

Inspection of TCP and UDP channels initiated from the router enables dynamic opening of pinholes on the interface access control list (ACL) to allow return traffic. You do not have to modify the ACL when a TCP connection such as Telnet is made from the router.

Inspection of local H.323 connections enables the deployment of CCME, H.323 gateway, and the Cisco IOS Firewall on the same router. This also simplifies ACL configuration on CCME’s interface through which H.323 connections are made. Before this feature, in addition to configuring ACLs to allow H.323 connections on a standard port (for example, port 1720), you had to configure ACLs to allow all dynamically negotiated data and media channels. With this feature you just configure the ACLs to allow H.323 control channels on port 1720. The Cisco IOS Firewall inspects all the traffic on the control channel and opens pinholes to allow dynamically negotiated data and media channels.

To enable Inspection of Router-Generated Traffic, specify the **router-traffic** keyword in the **ip inspect name** command of the appropriate protocol. This allows inspection of traffic to the router and the traffic passing through the router..

How to Configure Inspection of Router-Generated Traffic

Configuring H.323 Inspection

To configure the H.323 protocol, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-nam* e {**TCP** | **UDP** | **H323**} [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}][**router-traffic**][**timeout** *seconds*]
4. **interface** *type slot/port*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect name** *inspection-nam* e {**TCP** | **UDP** | **H323**} [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}][**router-traffic**][**timeout** *seconds*] **Example:** `Router(config)# ip inspect name test H.323 router-traffic`	Defines a set of inspection rules.
Step 4	**interface** *type* *slot/port* **Example:** `Router(config)# interface FE 0/0`	Configures an interface type.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Configuring CBAC

To configure CBAC, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *access-list-number* {**deny** | **permit**} *source*[*source-wildcard*] [**log**]
4. **ip inspect name** *inspection-nam* e {**TCP** | **UDP** | **H323**} [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}][**router-traffic**][**timeout** *seconds*]
5. **interface** *type slot/port*
6. **ip inspect** *inspection-name* {**in** | **out**}
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* {**deny** | **permit**} *source*[*source-wildcard*] [**log**] **Example:** `Router(config)# access-list 121 permit tcp host 100.168.11.1 any eq 1720`	Defines a standard IP access list.
Step 4	**ip inspect name** *inspection-nam* e {**TCP** | **UDP** | **H323**} [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}][**router-traffic**][**timeout** *seconds*] **Example:** `Router(config)# ip inspect name here H323 router-traffic timeout 180`	Defines a set of inspection rules.
Step 5	**interface** *type slot/port* **Example:** `Router(config)# Serial0/3/0`	Configures an interface type.

	Command or Action	Purpose

Step 6	**ip inspect** *inspection-name* {**in** | **out**}
	Example:
	`Router(config-if)# ip inspect test in`
		Enables the Cisco IOS Firewall on an interface.

Step 7	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Verifying the CBAC Configuration

To verify the CBAC configuration, perform the following task.

SUMMARY STEPS

1. **show ip inspect name** *inspection-name*
2. **show ip inspect config**
3. **show ip inspect interfaces**
4. **show ip inspect session detail**
5. **show ip inspect all**

DETAILED STEPS

Step 1 **show ip inspect name** *inspection-name*
Use this command to show a particular configured inspection rule. The following example configures the inspection rule myinspectionrule. The output shows the protocols that should be inspected by CBAC and the corresponding idle timeouts for each protocol.

Example:

```
Router# show ip inspect name myinspectionrule
Inspection Rule Configuration
 Inspection name myinspectionrule
  tcp timeout 3600
  udp timeout 30
  ftp timeout 3600
```

Step 2 **show ip inspect config**
Use this command to show the CBAC configuration, including global timeouts, thresholds, and inspection rules.

Example:

```
Router# show ip inspect config

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
```

Step 3 **show ip inspect interfaces**

Use this command to show the interface configuration with respect to applied inspection rules and access lists.

Example:

```
Router# show ip inspect interfaces

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
```

Step 4 **show ip inspect session   detail**

Use this command to display existing sessions that CBAC is currently tracking and inspecting. The following sample output shows that an outgoing ACL and an inbound ACL (dynamic ACLs) have been created to allow return traffic.

Example:

```
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
```

Step 5 **show ip inspect all**

Use this command to show all CBAC configuration and all existing sessions that are currently being tracked and inspected by CBAC.

Example:

```
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
```

Configuration Examples for Inspection of Router-Generated Traffic

Example Configuring CBAC with Inspection of H.323 Traffic

These commands create the ACL. In this example, TCP traffic from subnet 100.168.11.1, 192.168.11.50, and 192.168.100.1 is permitted.

```
access-list 120 permit tcp host 100.168.11.1 any eq 1720
access-list 121 permit tcp host 192.168.11.50 host 100.168.11.1 eq 1720
access-list 121 permit tcp host 192.168.100.1 host 100.168.11.1 eq 1720
```

These commands create the CBAC inspection rule LOCAL-H323, allowing inspection of the protocol traffic specified by the rule. This inspection rule sets the timeout value to 180 seconds for each protocol (except for RPC). The timeout value defines the maximum time that a connection for a given protocol can remain active without any traffic passing through the router. When these timeouts are reached, the dynamic ACLs that are inserted to permit the returning traffic are removed, and subsequent packets (possibly even valid ones) are not permitted.

```
ip inspect name LOCAL-H323 tftp timeout 180
ip inspect name LOCAL-H323 h323 router-traffic timeout 180
```

These commands apply the inspection rule and ACL. In this example, the inspection rule LOCAL-H323 is applied to traffic at interface Serial0/3/0.

```
interface Serial0/3/0
 ip address 11.168.11.2 255.255.255.0
 ip access-group 121 in
 ip access-group 120 out
 ip inspect LOCAL-H323 in
 ip inspect LOCAL-H323 out
 encapsulation frame-relay
frame-relay map ip 11.168.11.1 168 broadcast
 no frame-relay inverse-arp
 frame-relay intf-type dce
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
CBAC	*Cisco IOS Security Command Reference* "Configuring Context-Based Access Control"
H.323	*Cisco IOS H.323 Configuration Guide*

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
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Inspection of Router-Generated Traffic

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 16: Feature Information for Inspection of Router-Generated Traffic

Feature Name	Releases	Feature Information
Inspection of Router-Generated Traffic	12.3(14)T	The Inspection of Router-Generated Traffic feature allows Context-Based Access Control (CBAC) to inspect traffic that is originated by or destined to the router on which CBAC is configured. Previously, inspection of Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and H.323 connections initiated by or destined to the router were allowed.

CHAPTER 12

Transparent Cisco IOS Firewall

The Transparent Cisco IOS Firewall feature allows users to “drop” a Cisco IOS Firewall in front of their existing network without changing the statically defined IP addresses of their network-connected devices. Thus, users can allow selected devices from a subnet to traverse the firewall while access to other devices on the same subnet is denied.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Transparent Cisco IOS Firewall

Layer 3 IP Packet Support Only

Only IP packets (TCP, User Datagram Protocol [UDP] , and Internet Control Message Protocol [ICMP]) are subjected to inspection by the transparent firewall. Non-IP traffic is bridged as usual without interference from the transparent firewall. However, if users wish to block non-IP traffic, the MAC access control lists (ACLs) can be applied on interfaces to filter out non-IP traffic and allow only IP traffic.

The following example shows how to configure an ACL that permits all IP packets (0x0800) into the Ethernet interface but denies all Internetwork Packet Exchange (IPX) packets (0x8137):

```
Router(config)# access-list 201 permit 0x0800
Router(config)# access-list 201 deny 0x8137
Router(config)# interface ethernet 0
Router(config-if)# bridge-group 1 input-type-list 201
```

VLAN Trunk Bridging

Bridging between VLAN trunks works only for dot1q encapsulation; Inter-Switch Link (ISL) encapsulation will not work. (However, ISL VLANs will work if subinterfaces are created and placed in a bridge group.)

Information About Transparent Cisco IOS Firewall

Benefit of the Transparent Firewall

Added Security with Minimum Configuration

Users can simply drop a transparent Cisco IOS Firewall into an existing network without having to reconfigure their statically defined devices. Thus, the tedious and costly overhead that is required to renumber devices on the trusted network is eliminated.

Transparent Firewall Overview

A typical Cisco IOS Firewall is a Layer 3 device with trusted and untrusted interfaces on different IP subnets. A Layer 3 firewall works well with Cisco IOS devices that function as routers with preexisting subnet separations. However, when a Layer 3 firewall is placed in an existing network, the network IP addresses must be reconfigured to accommodate the firewall.

A transparent Cisco IOS firewall acts as a Layer 2 transparent bridge with context-based access control (CBAC) and ACLs configured on the bridged interface. Because the Layer 2 firewall intercepts packets at Layer 2 and is “transparent” to the existing network, Layer 3 firewall limitations are not applicable.

Transparent Bridging Overview

If configured for bridging, a Cisco IOS device can bridge any number of interfaces. The device can complete basic bridging tasks such as learning MAC addresses on ports to restrict collision domains and running Spanning Tree Protocol (STP) to prevent looping in the topology.

Within bridging, a user can configure Integrated Routed Bridging (IRB), which allows a device to bridge on some interfaces while a Layer 3 Bridged Virtual Interface (BVI) is presented for routing. The bridge can determine whether the packet is to be bridged or routed on the basis of the destination of the Layer 2 or Layer 3 IP address in the packet. Configured with an IP address, the BVI can manage the device even if there is no interface configured for routing.

Layer 2 and Layer 3 Firewalls Configured on the Same Router

A transparent firewall supports a BVI for routing, so a packet that comes in on a bridged interface can be bridged or routed out of the BVI. This functionality allows a Layer 2 (transparent) firewall and a Layer 3 firewall to be configured on the same router: The transparent firewall operates on the bridged packets while the "normal" firewall operates on the routed packets. For example, if you have six interfaces on your router and two of them are in a bridge group, you can simultaneously configure and run normal inspection on the remaining four interfaces.

How to Configure a Transparent Cisco IOS Firewall

You configure a transparent firewall just as you would configure a Layer 3 firewall (via the **ip inspect** command, which can be configured on any of the bridged interfaces for the transparent firewall). Also, you configure transparent bridging for a firewall just as you would for any other Cisco IOS device.

Configuring a Bridge Group

Perform this task to configure a bridge group and to associate interfaces or subinterfaces in the configured bridge group.

Before You Begin

- If a BVI is not configured, you must disable IP routing (via the **no ip routing** command) for the bridging operation to take effect.
- If configured, a BVI must be configured with an IP address in the same subnet.
- You must configure a BVI if more than two interfaces are placed in a bridge group.

Note

- If more than two interfaces are assigned to a bridge group, any routers that are acting as first-hop gateways to hosts that are in the bridged network (the bridge group) must allow ICMP time-to-live (TTL) exceeded messages to pass.
- Spanning Tree Bridge Protocol Data Units (BPDU) and packets that are to be routed out of the bridge, if IRB is configured, are not inspected.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bridge** *bridge-group* **protocol** {**dec** | **ibm** | **ieee** | **vlan-bridge**
4. **interface** *type number*
5. **bridge-group** *bridge-group*
6. **exit**
7. **bridge irb**
8. **bridge** *bridge-group* **route** *protocol*
9. **interface** *type number*
10. **ip address** *ip-address mask*
11. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**bridge** *bridge-group* **protocol** {**dec** | **ibm** | **ieee** | **vlan-bridge** **Example:** `Router(config)# bridge 1 protocol ieee`	Defines the type of Spanning Tree Protocol (STP).
Step 4	**interface** *type number* **Example:** `Router(config)# interface Ethernet0`	Configures an interface type and enters interface configuration mode.
Step 5	**bridge-group** *bridge-group* **Example:** `Router(config-if)# bridge-group 1`	Assigns each network interface to a bridge group. **Note** Complete Step 4 and Step 5 for each interface you want to assign to a bridge group. **Note** You can also assign subinterfaces to a bridge group to control bridging between VLANs.

	Command or Action	Purpose
Step 6	**exit**	Exits interface configuration mode.
Step 7	**bridge irb** **Example:** `Router(config)# bridge irb`	Enables the Cisco IOS software to route a given protocol between routed interfaces and bridge groups or to route a given protocol between bridge groups. **Note** Step 7 through Step 11 are necessary only if you want to configure a BVI.
Step 8	**bridge** *bridge-group* **route** *protocol* **Example:** `Router(config)# bridge 1 route ip`	Enables the routing of a specified protocol in a specified bridge group.
Step 9	**interface** *type number* **Example:** `Router(config)# interface BVI1`	Configures a BVI and enters interface configuration mode.
Step 10	**ip address** *ip-address mask* **Example:** `Router(config-if) ip address 10.1.1.1 255.255.255.0`	Sets a primary IP address for an interface.
Step 11	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Restarts a disabled interface.

Examples

The following example shows how to configure interfaces "ethernet0" and "ethernet1" in a bridge group. These interfaces are associated with the BVI interface "BVI1," which can be reached from any host on either of the interfaces via the IP address 10.1.1.1.

```
Router(config)# bridge 1 protocol ieee
Router(config)# interface ethernet0
Router(config-if)# bridge-group 1
Router(config-if)# interface ethernet1
Router(config-if)# bridge-group 1
Router(config-if)# exit
! Configure the BVI.
Router(config)# bridge irb
Router(config)# bridge 1 route ip
Router(config)# interface BVI1
Router(config-if)# ip address 10.1.1.1 255.255.255.0
Router(config-if)# no shutdown
```

Troubleshooting Tips

To display the status of each bridge group, use the **show bridge-group** command or to display entries in the bridge table, use the **show bridge** command.

What to Do Next

After you have configured the bridge group, you must configure an inspection rule and at least one IP ACL. To complete this task, refer to the following section, "Configuring Inspection and ACLs."

Note If inspection is not configured on any interface in the bridge group, IP ACLs on bridged interfaces will not be active.

Configuring Inspection and ACLs

Use this task to configure an inspection rule and apply it on the appropriate interface. Also, use this task to configure at least one ACL and apply it on one or more of the interfaces that you configured in the bridge group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}]
4. **interface** *type number*
5. **ip inspect** *inspection-name* {**in** | **out**}
6. **exit**
7. **access-list** *access-list-number* {**permit** | **deny**} {*type-code wild-mask*| *address mask*}
8. **interface** *type number*
9. **ip access-group** {*access-list-number* | *access-list-name*} **in** | **out**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect name** *protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] **Example:** [**timeout** *seconds*] **Example:** `Router(config)# ip inspect name test tcp`	Defines a set of inspection rules.
Step 4	**interface** *type number* **Example:** `Router(config)# interface Ethernet0`	Configures an interface type and enters interface configuration mode.
Step 5	**ip inspect** *inspection-name* {**in** | **out**} **Example:** `Router(config-if)#` **ip inspect test in**	Applies a set of inspection rules to an interface.
Step 6	**exit**	Exits interface configuration mode.

Step 7	**access-list** *access-list-number* {**permit** | **deny**} {*type-code wild-mask*| *address mask*}
	Example:
	`Router(config)#`
	`access-list 156 permit 10.1.1.0 0.0.0.255 any`
		Configures the ACL.
		Note Repeat this step for each ACL that you want to configure.

Step 8	**interface** *type number* **Example:** `Router(config)# interface Ethernet0`	Configures an interface type and enters interface configuration mode. **Note** Repeat Steps 8 and 9 for each ACL that you want to apply to inbound packets from a specific interface.
Step 9	**ip access-group** {*access-list-number* | *access-list-name*} **in** | **out**	Controls access to an interface.

	Command or Action	Purpose

	Example:
	`Router(config-if)`
	` ip access-group 156 in`	

Examples

The following example shows how to configure an inspection rule on interface "ethernet0," which is the inside interface. Policies can be specified via ACL 156 or 101; for example, ACL 156 can be used to specify that rlogin and rsh are not allowed for the internal users, and ACL 101 can be used to specify that an external host requires connectivity to a particular host in the internal domain.

```
Router(config)# ip inspect name test tcp
Router(config)# interface ethernet0
Router(config-if)# ip inspect test in
Router(config-if)# exit
!
! Configure the ACLs.
Router(config)# access-list 101 deny ip any any
Router(config)# access-list 156 permit 10.1.1.0 0.0.0.255 any
Router(config)# access-list 156 deny ip any any
Router(config)# interface ethernet0
Router(config-if) ip access-group 156 in
Router(config)# interface ethernet1
Router(config-if) ip access-group 101 in
```

Forwarding DHCP Traffic

Use this task to enable a transparent firewall to forward DHCP packets across the bridge without inspection; that is, the **ip inspect L2-transparent dhcp-passthrough** command overrides the ACL for DHCP packets, so DHCP packets will be forwarded even if the ACL is configured to deny all IP packets. Thus, clients on one side of the bridge can get an IP address from a DHCP server on the opposite side of the bridge.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect L2-transparent dhcp-passthrough**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip inspect L2-transparent dhcp-passthrough** **Example:** `Router#(config) ip inspect L2-transparent dhcp-passthrough`	Allows a transparent firewall to forward DHCP passthrough traffic.

Monitoring Transparent Firewall Events

Use either of these optional steps to monitor the activity of the transparent firewall.

Note Effective with Cisco IOS Release 12.4(20)T, the **debug ip inspect** command is replaced by the **debug policy-firewall** command. See the Cisco IOS Debug Command Reference for more information.

SUMMARY STEPS

1. **enable**
2. **debug ip inspect L2-transparent packet | dhcp-passthrough**
3. **show ip inspect** {**name** *inspection-name*| **config** | **interfaces** | **session** [**detail**] | **all**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ip inspect L2-transparent packet | dhcp-passthrough**	Enables debugging messages for transparent firewall events.

	Command or Action	Purpose
	Example: `Router# debug ip inspect L2-transparent dhcp-passthrough`	• **packet** --Displays messages for all debug packets that are inspected by the transparent firewall. • **dhcp-passthrough--** Displays debug messages only for DHCP pass-through traffic that the transparent firewall forwards across the bridge.

Step 3	**show ip inspect** {**name** *inspection-name*| **config** | **interfaces** | **session** [**detail**] | **all**}
	Example:
	`Router#`
	`show ip inspect all`
		Displays Cisco IOS Firewall configuration and session information.
		• If the transparent firewall is configured, use the **all** keyword to display the bridging interface in the interface configuration section of the output.

Examples

The following sample output is a portion of the **show ip inspect all** command that shows the bridging interface:

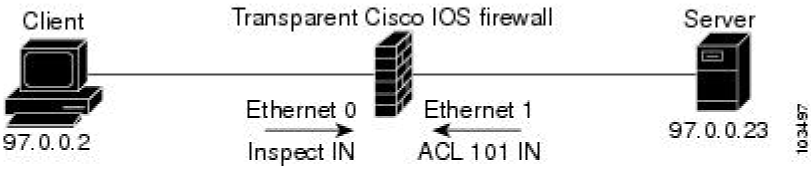
```
Router# show ip inspect all
.
.
.
Interface Configuration
! Below is the bridging interface.
Interface Ethernet1
Inbound inspection rule is test
tcp alert is on audit-trail is off timeout 3600
ftp alert is on audit-trail is off timeout 3600
Outgoing inspection rule is not set
Inbound access list is not set
Outgoing access list is 156
.
.
.
```

Configuration Examples for Transparent Cisco IOS Firewall

Example Comprehensive Transparent Firewall Configuration

The following example and sample topology (see the figure below) illustrate how to configure and debug a transparent Cisco IOS Firewall configuration between a client, a firewall, and a server. This example also includes **show** command output for additional configuration verification. After you have configured a transparent firewall, you can Telnet from the client to the server through the firewall. (See the section “Example Monitoring Telnet Connections via debug and show Output.”

Figure 10: Sample Topology for Transparent Firewall Configuration



Note Effective with Cisco IOS Release 12.4(20)T, the **debug ip inspect** command is replaced by the **debug policy-firewall** command. See the Cisco IOS Debug Command Reference for more information.

```
! Enable debug commands.
Router# debug ip inspect L2-transparent packet
INSPECT L2 firewall debugging is on
Router# debug ip inspect object-creation
INSPECT Object Creations debugging is on
Router# debug ip inspect object-deletion
INSPECT Object Deletions debugging is on
! Start the transparent firewall configuration process
Router# config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
! Configure bridging
Router(config)# bridge 1 protocol ieee
Router(config)# bridge irb
Router(config)# bridge 1 route ip
Router(config)# interface bvi1
*Mar  1 00:06:42.511:%LINK-3-UPDOWN:Interface BVI1, changed state to down.
Router(config-if)# ip address 209.165.200.225 255.255.255.254
! Configure inspection
Router(config)# ip inspect name test tcp
! Following debugs show the memory allocated for CBAC rules.
*Mar  1 00:07:21.127:CBAC OBJ_CREATE:create irc 817F04F0 (test)
*Mar  1 00:07:21.127:CBAC OBJ_CREATE:create irt 818AED20 Protocol:tcp Inactivity time:0
test
Router(config)# ip inspect name test icmp
Router(config)#
*Mar  1 00:07:39.211:CBAC OBJ_CREATE:create irt 818AEDCC Protocol:icmp Inactivity time:0
! Configure Bridging on ethernet0 interface
Router(config)# interface ethernet0
Router(config-if)# bridge-group 1
*Mar  1 00:07:49.071:%LINK-3-UPDOWN:Interface BVI1, changed state to up
*Mar  1 00:07:50.071:%LINEPROTO-5-UPDOWN:Line protocol on Interface BVI1, changed state to
 up
! Configure inspection on ethernet0 interface
Router(config-if)# ip inspect test in
Router(config-if)#
*Mar  1 00:07:57.543:CBAC OBJ_CREATE:create idbsb 8189CBFC (Ethernet0)
! Incremented the number of bridging interfaces configured for inspection
*Mar  1 00:07:57.543:L2FW:Incrementing L2FW i/f count
Router(config-if)# interface ethernet1
! Configure bridging and ACL on interface ethernet1
Router(config-if)# bridge-group 1
Router(config-if)# ip access-group 101 in
*Mar  1 00:08:26.711:%LINEPROTO-5-UPDOWN:Line protocol on Interface Ethernet1, changed state
 to up
Router(config-if)# end
Router(config)# end
!
! Issue the show running-config command to verify the complete transparent firewall !
```

```
configuration.
Router# show running-config
Building configuration...
Current configuration :1126 bytes
!
version 12.3
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname Firewall
!
logging buffered 12000 debugging
no logging console
!
no aaa new-model
ip subnet-zero
no ip domain lookup
!
!
ip inspect name test tcp
ip inspect name test icmp
ip audit notify log
ip audit po max-events 100
no ftp-server write-enable
!
!
!
no crypto isakmp enable
!
!
bridge irb
!
!
interface Ethernet0
 no ip address
 no ip proxy-arp
 ip inspect test in
 bridge-group 1
 hold-queue 100 out
!
interface Ethernet1
 no ip address
 ip access-group 101 in
 no ip unreachables
 no ip proxy-arp
 duplex auto
 bridge-group 1
!
interface BVI1
 ip address 209.165.200.225 255.255.255.254
!
ip classless
ip route 9.1.0.0 255.255.0.0 9.4.0.1
no ip http server
no ip http secure-server
!
!
ip access-list log-update threshold 1
access-list 101 permit icmp any any log
access-list 101 deny   ip any any log
!
control-plane
!
bridge 1 protocol ieee
bridge 1 route ip
!
line con 0
 no modem enable
 stopbits 1
line aux 0
 stopbits 1
```

```
line vty 0 4
 login
!
scheduler max-task-time 5000
!
end
!
! Issue show brige commands to check the tables.
Router# show bridge
Total of 300 station blocks, 300 free
Codes:P - permanent, S - self
Bridge Group 1:
! The bridge table is empty because no traffic has been seen
!
Router# show bridge group
Bridge Group 1 is running the IEEE compatible Spanning Tree protocol
Port 2 (Ethernet0) of bridge group 1 is forwarding
Port 3 (Ethernet1) of bridge group 1 is forwarding
! Note that the interfaces are in a "forwarding" state. The interfaces move from  ! a listening
 state to a learning state and finally to a forwarding state. It takes ! approximately 30
seconds to move to a forwarding after "bridge-group 1" is configured.
```

Example Monitoring Telnet Connections via debug and show Output

The following examples shows how to monitor established Telnet connections from the client to the server through the firewall (see the figure above) and from the server to the client. In these example, the **debug ip inspect L2-transparent packet** command has been issued to generate the debug messages. Relevant **show** commands are also issued for additional verification.

Note Effective with Cisco IOS Release 12.4(20)T, the **debug ip inspect** command is replaced by the **debug policy-firewall** command. See the Cisco IOS Debug Command Reference for more information.

Telnet Connection from the Client (97.0.0.2) to the Server (97.0.0.23)

The following example is output from the initial Telnet connection between the client and the server. A subsequent connection is established to highlight differences in the debug output. Explanations are given inline.

```
! A packet is received by the firewall in the flood path because the bridge-table is !
initially empty. However, the client seems to have the server's mac-address in its ARP !
cache, so the bridge floods the packet and it appears in the firewall's "flood" path.
*Mar  1 00:17:32.119:L2FW:insp_l2_flood:input is Ethernet0 output is Ethernet1
! Source and destination IP addresses and the L4 protocol of the packet
*Mar  1 00:17:32.123:L2FW*:Src 97.0.0.2 dst 97.0.0.23 protocol tcp
! ACL processing status. An ACL is not configured in this direction; that is, from the !
client to the server.
*Mar  1 00:17:32.123:L2FW:Input ACL not configured or the ACL is bypassed
*Mar  1 00:17:32.123:L2FW:Output ACL is not configured or ACL is bypassed
! If there are exactly two interfaces in the bridge-group and the packet is in flood path,
 ! the firewall invokes inspection directly, skipping the Unicast flood algorithm. If there
 ! are more than 2 interfaces, the firewall "drops" the packet and issues the algorithm.
*Mar  1 00:17:32.123:L2FW:FLOOD number of i/fs in bridge-group is exactly 2. Calling
Inspection
! The packet is being inspected.
*Mar  1 00:17:32.123:L2FW:insp_l2_inspection:input is Ethernet0 output is Ethernet1
*Mar  1 00:17:32.123:L2FW*:Src 97.0.0.2 dst 97.0.0.23 protocol tcp
*Mar  1 00:17:32.123:L2FW:Input ACL not configured or the ACL is bypassed
*Mar  1 00:17:32.123:L2FW:Output ACL is not configured or ACL is bypassed
! Memory is allocated for the transparent firewall attributes in the session structure
```

```
*Mar  1 00:17:32.123:L2FW:allocating L2 extension for sis
! CBAC-related debug messages: The packet has been passed to the existing CBAC code.
*Mar  1 00:17:32.123:CBAC Pak 814635DC sis 816C9C24 initiator_addr (97.0.0.2:11016)
responder_addr (97.0.0.23:23)
initiator_alt_addr (97.0.0.2:11016) responder_alt_addr (97.0.0.23:23)
! CBAC session structure has been allocated
*Mar  1 00:17:32.127:CBAC OBJ_CREATE:create sis 816C9C24
*Mar  1 00:17:32.127:CBAC OBJ-CREATE:sid 816D69D8 acl 101 Prot:tcp
*Mar  1 00:17:32.127: Src 97.0.0.23 Port [23:23]
*Mar  1 00:17:32.127: Dst 97.0.0.2 Port [11016:11016]
! The Layer 2 header length is being computed for caching the L2 header, which will be !
used if a TCP RST should be sent in the future to tear down the connection.
*Mar  1 00:17:32.127:L2FW:L2 header length(initiator->responder) is 14
! Checks to see if the header is 802.3, SNAP, SAP. (This header is 802.3.)
*Mar  1 00:17:32.127:L2FW:info_start is NULL for init->rsp
*Mar  1 00:17:32.127:CBAC OBJ_CREATE:create host entry 816D4018 addr 97.0.0.23 bucket 118
! CBAC has indicated that the packet should be passed
*Mar  1 00:17:32.127:L2FW:insp_inspection returned FALSE. PASS
! The next packet in the flow has arrived on the interrupt path. This packet is from the !
 server (ethernet1) to the client (ethernet0).
*Mar  1 00:17:32.131:L2FW*:insp_l2_fast_inspection:pak 812C9084, input-interface Ethernet1,
 output-interface Ethernet0
*Mar  1 00:17:32.131:L2FW*:Src 97.0.0.23 dst 97.0.0.2 protocol tcp
*Mar  1 00:17:32.131:L2FW:Input ACL not configured or the ACL is bypassed
*Mar  1 00:17:32.131:L2FW:Output ACL is not configured or ACL is bypassed
! The Layer 2 header length is computed and will be cached
*Mar  1 00:17:32.131:L2FW:L2 header length is 14 (rsp->init)
*Mar  1 00:17:32.131:L2FW:info_start is NULL rsp->init
! CBAC has indicated that the packet should be forwarded
*Mar  1 00:17:32.131:L2FW*:insp_l2_fast_inspection returning INSP_L2_OK
! A new packet has arrived from the client. The following trace repeats for each packet
received by the firewall
*Mar  1 00:17:32.135:L2FW*:insp_l2_fast_inspection:pak 81462FB4, input-interface Ethernet0,
 output-interface Ethernet1
*Mar  1 00:17:32.135:L2FW*:Src 97.0.0.2 dst 97.0.0.23 protocol tcp
*Mar  1 00:17:32.135:L2FW:Input ACL not configured or the ACL is bypassed
*Mar  1 00:17:32.135:L2FW:Output ACL is not configured or ACL is bypassed
*Mar  1 00:17:32.135:L2FW*:insp_l2_fast_inspection returning INSP_L2_OK
       ...<more packets >...
! The host entry for the server is deleted.
*Mar  1 00:17:32.263:CBAC OBJ_DELETE:delete host entry 816D4018 addr 97.0.0.23
! Issue the show ip inspect command to verify that a CBAC session has been established
Router# show ip inspect session detailed
Established Sessions
 Session 816C9C24 (97.0.0.2:11016)=>(97.0.0.23:23) tcp SIS_OPEN
  Created 00:00:28, Last heard 00:00:09
  Bytes sent (initiator:responder) [38:75]
  In  SID 97.0.0.23[23:23]=>97.0.0.2[11016:11016] on ACL 101 (12 matches)
Router#
!
! Issue the show bridge command to verify that entries for the client and server have been
 ! created in the bridge-table.
Router# show bridge
Total of 300 station blocks, 298 free
Codes:P - permanent, S - self
Bridge Group 1:
    Address       Action   Interface       Age   RX count   TX count
0008.a3b6.b603   forward   Ethernet0         2         14         12
0007.0d97.308f   forward   Ethernet1         2         12         13
Router#
!
! Close the TCP connection (by typing exit at the client).
*Mar  1 00:21:26.259:CBAC OBJ_DELETE:delete sis 816C9C24
*Mar  1 00:21:26.259:CBAC OBJ-DELETE:sid 816D69D8 on acl 101 Prot:tcp
*Mar  1 00:21:26.259: Src 97.0.0.23 Port [23:23]
*Mar  1 00:21:26.259: Dst 97.0.0.2 Port [11016:11016]
! The data structures pertaining to the Layer 2 firewall have been deleted from the !
session. The session has also been deleted.
*Mar  1 00:21:26.259:L2FW:Deleting L2FW data structures
```

A New Telnet Connection from the Client (97.0.0.2) to the Server (97.0.0.23)

```
! The initial SYN packet from the client has arrived in the interrupt path. Note that the
! corresponding packet from the previous telnet session came in on the flood path because
! the bridge-table was empty so the bridge was forced to flood the packet. Since the !
bridge-table is now populated, the packet does not not to be flooded. This is the only !
difference between the previous telnet session and this session. Subsequent packets will !
 follow the same path (and generate the same debugs) as the previous session.
*Mar  1 00:23:31.883:L2FW*:insp_l2_fast_inspection:pak 81465190, input-interface Ethernet0,
 output-interface Ethernet1
*Mar  1 00:23:31.883:L2FW*:Src 97.0.0.2 dst 97.0.0.23 protocol tcp
*Mar  1 00:23:31.883:L2FW:Input ACL not configured or the ACL is bypassed
*Mar  1 00:23:31.883:L2FW:Output ACL is not configured or ACL is bypassed
! CBAC has indicated that the packet should be punted to the process path since memory !
allocation and the control-plane is involved
*Mar  1 00:23:31.883:L2FW*:insp_l2_fast_inspection returning INSP_L2_PUNT
! After being punted from the interrupt path, the packet has arrived at the process level
! for inspection. Moving forward, the debug messages are similar to the flood case in the
! previous session.
*Mar  1 00:23:31.883:L2FW:insp_l2_inspection:input is Ethernet0 output is Ethernet1
*Mar  1 00:23:31.883:L2FW*:Src 97.0.0.2 dst 97.0.0.23 protocol tcp
*Mar  1 00:23:31.883:L2FW:Input ACL not configured or the ACL is bypassed
*Mar  1 00:23:31.883:L2FW:Output ACL is not configured or ACL is bypassed
*Mar  1 00:23:31.887:L2FW:allocating L2 extension for sis
*Mar  1 00:23:31.887:CBAC Pak 81465190 sis 816C9C24 initiator_addr (97.0.0.2:11017)
responder_addr (97.0.0.23:23)
initiator_alt_addr (97.0.0.2:11017) responder_alt_addr (97.0.0.23:23)
*Mar  1 00:23:31.887:CBAC OBJ_CREATE:create sis 816C9C24
*Mar  1 00:23:31.887:CBAC OBJ-CREATE:sid 816D69D8 acl 101 Prot:tcp
*Mar  1 00:23:31.887: Src 97.0.0.23 Port [23:23]
*Mar  1 00:23:31.887: Dst 97.0.0.2 Port [11017:11017]
*Mar  1 00:23:31.887:L2FW:L2 header length(initiator->responder) is 14
*Mar  1 00:23:31.887:L2FW:info_start is NULL for init->rsp
*Mar  1 00:23:31.887:CBAC OBJ_CREATE:create host entry 816D4018 addr 97.0.0.23 bucket 118
! CBAC has indicated that the packet should be Passed
*Mar  1 00:23:31.891:L2FW:insp_inspection returned FALSE. PASS
!
! Issue the show ip inspect command to verify the newly created inspect session
Router# show ip inspect session details
Established Sessions
 Session 816C9C24 (97.0.0.2:11017)=>(97.0.0.23:23) tcp SIS_OPEN
  Created 00:00:52, Last heard 00:00:37
  Bytes sent (initiator:responder) [38:75]
  In  SID 97.0.0.23[23:23]=>97.0.0.2[11017:11017] on ACL 101 (10 matches)
Router#
```

Telnet Connection from the Server (97.0.0.23) to the Client (97.0.0.2)

The following sample output is from a Telnet connection that was initiated from the server to the client. This connection will not go through because “ACL 101” is configured to allow only ICMP packets and deny all other packets. Note that inspection is not configured from the server to the client. This example is shown to display the debug messages that are associated with dropped packets.

```
! The first packet from the server comes in on ethernet1 interface
*Mar  1 00:26:12.367:L2FW*:insp_l2_fast_inspection:pak 815C89FC, input-interface Ethernet1,
 output-interface Ethernet0
*Mar  1 00:26:12.367:L2FW*:Src 97.0.0.23 dst 97.0.0.2 protocol tcp
! This packet is punted up since ACL 101 is configured for logging. Logging happens in the
 process path. If logging was not configured, the packet would have been dropped instead
of being punted to process level
*Mar  1 00:26:12.367:L2FW:Packet punted up by Input ACL for logging
! The packet arrives at process level
*Mar  1 00:26:12.367:L2FW:insp_l2_inspection:input is Ethernet1 output is Ethernet0
*Mar  1 00:26:12.371:L2FW*:Src 97.0.0.23 dst 97.0.0.2 protocol tcp
! The ACL log is generated
*Mar  1 00:26:12.371:%SEC-6-IPACCESSLOGP:list 101 denied tcp 97.0.0.23(11045) -> 97.0.0.2(23),
 1 packet
```

```
! The packet is dropped by the ACL
*Mar  1 00:26:12.371:L2FW:Packet processed and dropped by Input ACL
! The packet is dropped by the ACL and is therefore NOT sent to CBAC for inspection
*Mar  1 00:26:12.371:L2FW:Packet is dropped in insp_l2_inspection
```

Examples Configuring and Verifying DHCP Pass-Through Traffic

The following examples show how to verify (via debug messages) DHCP pass-through that has been allowed and traffic that has not been allowed.

Example Allowing DHCP Pass-Through Traffic

In this example, the static IP address of the client is removed and the address is acquired via DHCP using the **ip address dhcp** command on the interface that is connected to the transparent firewall.

```
Router# show debug
ARP:
  ARP packet debugging is on
L2 Inspection:
  INSPECT L2 firewall debugging is on
  INSPECT L2 firewall DHCP debugging is on
Router#
Router#
! Configure DHCP passthrough
Router(config)# ip insp L2-transparent dhcp-passthrough
! The DHCP discover broadcast packet arrives from the client. Since this packet is a !
broadcast (255.255.255.255), it arrives in the flood path
*Mar  1 00:35:01.299:L2FW:insp_l2_flood:input is Ethernet0 output is Ethernet1
*Mar  1 00:35:01.299:L2FW*:Src 0.0.0.0 dst 255.255.255.255 protocol udp
*Mar  1 00:35:01.299:L2FW:udp ports src 68 dst 67
*Mar  1 00:35:01.299:L2FW:src 0.0.0.0 dst 255.255.255.255
! The DHCP pass through flag is checked and the packet is allowed
*Mar  1 00:35:01.299:L2FW:DHCP packet seen. Pass-through flag allows the packet
! The packet is a broadcast packet and therefore not sent to CBAC
*Mar  1 00:35:01.299:L2FW*:Packet is broadcast or multicast.PASS
! The DHCP server 97.0.0.23 responds to the client's request
*Mar  1 00:35:01.303:L2FW:insp_l2_flood:input is Ethernet1 output is Ethernet0
*Mar  1 00:35:01.303:L2FW*:Src 97.0.0.23 dst 255.255.255.255 protocol udp
*Mar  1 00:35:01.307:L2FW:udp ports src 67 dst 68
*Mar  1 00:35:01.307:L2FW:src 97.0.0.23 dst 255.255.255.255
*Mar  1 00:35:01.307:L2FW:DHCP packet seen. Pass-through flag allows the packet
*Mar  1 00:35:01.307:L2FW*:Packet is broadcast or multicast.PASS
*Mar  1 00:35:01.311:L2FW:insp_l2_flood:input is Ethernet0 output is Ethernet1
*Mar  1 00:35:01.311:L2FW*:Src 0.0.0.0 dst 255.255.255.255 protocol udp
*Mar  1 00:35:01.311:L2FW:udp ports src 68 dst 67
*Mar  1 00:35:01.311:L2FW:src 0.0.0.0 dst 255.255.255.255
*Mar  1 00:35:01.315:L2FW:DHCP packet seen. Pass-through flag allows the packet
*Mar  1 00:35:01.315:L2FW*:Packet is broadcast or multicast.PASS
*Mar  1 00:35:01.315:L2FW:insp_l2_flood:input is Ethernet1 output is Ethernet0
*Mar  1 00:35:01.323:L2FW*:Src 97.0.0.23 dst 255.255.255.255 protocol udp
*Mar  1 00:35:01.323:L2FW:udp ports src 67 dst 68
*Mar  1 00:35:01.323:L2FW:src 97.0.0.23 dst 255.255.255.255
*Mar  1 00:35:01.323:L2FW:DHCP packet seen. Pass-through flag allows the packet
*Mar  1 00:35:01.323:L2FW*:Packet is broadcast or multicast.PASS
! The client has an IP address (97.0.0.5) and has issued a G-ARP to let everyone know it's
address
*Mar  1 00:35:01.327:IP ARP:rcvd rep src 97.0.0.5 0008.a3b6.b603, dst 97.0.0.5 BVI1
Router#
```

Example Denying DHCP Pass-Through Traffic

In this example, DHCP pass-through traffic is not allowed (via the **no ip inspect L2-transparent dhcp-passthrough**command). The client is denied when it attempts to acquire a DHCP address from the server.

```
! Deny DHCP pass-through traffic
Router(config)# no ip inspect L2-transparent dhcp-passthrough

! The DHCP discover broadcast packet arrives from the client
*Mar  1 00:36:40.003:L2FW:insp_l2_flood:input is Ethernet0 output is Ethernet1
*Mar  1 00:36:40.003:L2FW*:Src 0.0.0.0 dst 255.255.255.255 protocol udp
*Mar  1 00:36:40.003:L2FW:udp ports src 68 dst 67
*Mar  1 00:36:40.007:L2FW:src 0.0.0.0 dst 255.255.255.255
! The pass-through flag is checked
*Mar  1 00:36:40.007:L2FW:DHCP packet seen. Pass-through flag denies the packet
! The packet is dropped because the flag does not allow DHCP passthrough traffic. Thus, !
the client cannot acquire an address, and it times out
*Mar  1 00:36:40.007:L2FW:FLOOD Dropping the packet after ACL check.
Router#
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco IOS Firewall commands	*Cisco IOS Security Command Reference*
Bridging commands	*Cisco IOS Bridging and IBM Networking Command Reference, Volume 1 of 2: Bridging*
Additional bridging configuration information	*The section "Bridging" of the Cisco IOS Bridging and IBM Networking Configuration Guide*
DHCP configuration information	The chapter "Configuring DHCP" in the *Cisco IOS IP Configuration Guide*

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
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Transparent Cisco IOS Firewall

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Note The table below lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Table 17: Feature Information for Transparent Cisco IOS Firewall

Feature Name	Releases	Feature Information
Transparent Cisco IOS Firewall	12.3(7)T	The Transparent Cisco IOS Firewall feature allows users to “drop” a Cisco IOS Firewall in front of their existing network without changing the statically defined IP addresses of their network-connected devices. Thus, users can allow selected devices from a subnet to traverse the firewall while access to other devices on the same subnet is denied. The following commands were introduced or modified: **debug ip inspect L2-transparent**, **ip inspect L2-transparent dhcp-passthrough**.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.