Security Configuration Guide: Context-Based Access Control Firewall, Cisco IOS Release 15.2S

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

CONTENTS

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 feature information and caveats, see 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 document.

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:

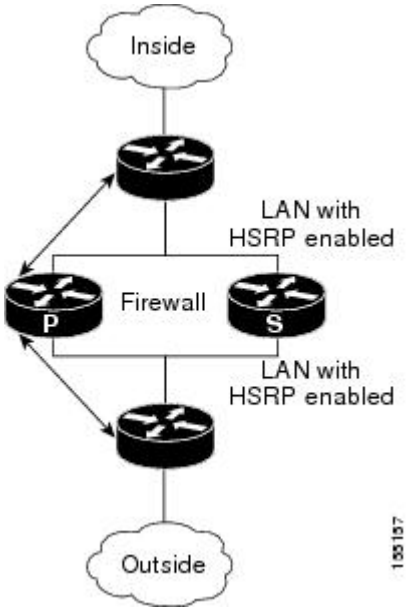
- Dual LAN Interface
- LAN WAN Interface

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 1 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, page 5
- Bulk Synchronization, page 5

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.

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** **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 *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.

	Command or Action	Purpose
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* **Example:** `Router(config-if)# standby 1 timers 1 5`	(Optional) Configures the time between hello packets and the time before other routers declare the active Hot Standby or standby router to be down. • *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, page 8
- What to Do Next, page 8

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.

- 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.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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, page 12
- What to Do Next, page 12

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 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.

	Command or Action	Purpose
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.

	Command or Action	Purpose
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** **Example:** `Router> enable`	Enables privileged EXEC mode. • 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.

Command or Action	Purpose
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.

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

The table below contains the stateful failover HA error messages.

Table 1 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 2 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 3 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**.

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 feature information and caveats, see 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 document.

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

CBAC has the following restrictions:

- CBAC is available only for IP protocol traffic. Only TCP and UDP packets are inspected. (Other IP traffic, such as ICMP, cannot be inspected with CBAC and should be filtered with basic access lists instead.)
- If you reconfigure your access lists when you configure CBAC, be aware that if your access lists 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.)
- Packets with the firewall as the source or destination address are not inspected by CBAC.
- CBAC ignores ICMP Unreachable messages.
- H.323 V2 and RTSP protocol inspection supports only the following multimedia client-server applications: Cisco IP/TV, RealNetworks RealAudio G2 Player, Apple QuickTime 4.
- When you configure an inspect rule on both the 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, the traffic is inspected on both the ingress and egress interfaces if the protocol is configured on both the ingress and egress interfaces and if the protocol is configured only on the egress interface.

You can use CBAC together with all the other firewall features mentioned previously in the “Cisco IOS Firewall Overview” chapter.

CBAC works with fast switching and process switching.

- FTP Traffic and CBAC, page 24
- IPSec and CBAC Compatibility, page 24

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, page 25
- What CBAC Does Not Do, page 27
- How CBAC Works-Overview, page 27
- How CBAC Works-Details, page 28
- When and Where to Configure CBAC, page 30
- The CBAC Process, page 30

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:

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

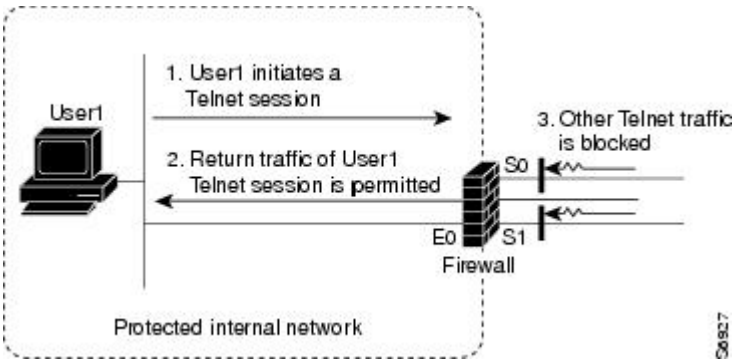
This command causes CBAC to inspect the packets coming into this interface from the network. If a packet is attempting to initiate a session, CBAC will then determine if this protocol is allowed, create a CBAC session, add the appropriate ACLs to allow return traffic and do any needed content inspection 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 2 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, page 32
- H.323 Support, page 33

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).

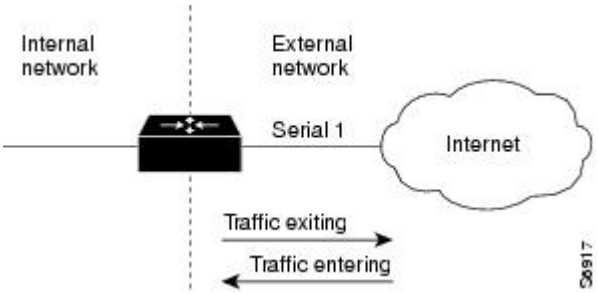
If you will be configuring CBAC in two directions, you should configure CBAC in one direction first, using the appropriate “internal” and “external” interface designations. When you configure CBAC in the other direction, the interface designations will be swapped. (CBAC can be configured in two directions 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.

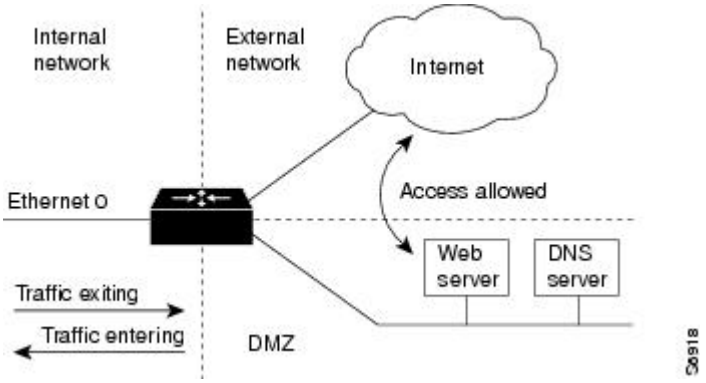
The first topology is shown in the figure below. In this simple topology, CBAC is configured for the external interface Serial 1. This prevents specified protocol traffic from entering the firewall and the internal network, unless the traffic is part of a session initiated from within the internal network.

Figure 3 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 4 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, page 35
- External Interface, page 36
- Internal Interface, page 37

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, page 40

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 4 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)

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.

Timeout or Threshold Value to Change	Command		Default
The length of time a UDP session will still be managed after no activity (the UDP idle timeout). 1	**time**	**ip inspect udp idle-** *seconds*	30 seconds
The length of time a DNS name lookup session will still be managed after no activity.	**timeout** *seconds*	**ip inspect dns-**	5 seconds
The number of existing half-open sessions that will cause the software to start deleting half-open sessions.[2]	**incomplete high**	**ip inspect max-** *number*	500 existing half-open sessions
The number of existing half-open sessions that will cause the software to stop deleting half-open sessions. 2	**incomplete low**	**ip inspect max-** *number*	400 existing half-open sessions
The rate of new sessions that will cause the software to start deleting half-open sessions. 2	**minute high**	**ip inspect one-** *number*	500 half-open sessions per minute
The rate of new sessions that will cause the software to stop deleting half-open sessions. 2	**minute low**	**ip inspect one-** *number*	400 half-open sessions per minute

2 See the following section, “Half-Open Sessions,” for more information.

Timeout or Threshold Value to Change	Command		Default
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]	**incomplete host**	**ip inspect tcp max-** *number* **block-time** *minutes*	50 existing half-open TCP sessions; 0 minutes

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, page 47
- Configuring Generic TCP and UDP Inspection, page 49

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, page 48
- Configuring Java Blocking, page 48

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.

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.) • • • Router(config)# **access-list***access-list-number* {**deny** | **permit**} *protocolsource* [*source-wildcard*]eq www *destination* [*destination-wildcard*]	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.
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.

Command	Purpose
`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.

	Command or Action	Purpose
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, page 52

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, page 54
- ATM Interface Configuration Example, page 54
- Remote Office to ISP Configuration Example, page 56
- Remote Office to Branch Office Configuration Example, page 58
- Two-Interface Branch Office Configuration Example, page 60
- Multiple-Interface Branch Office Configuration Example, page 63

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.

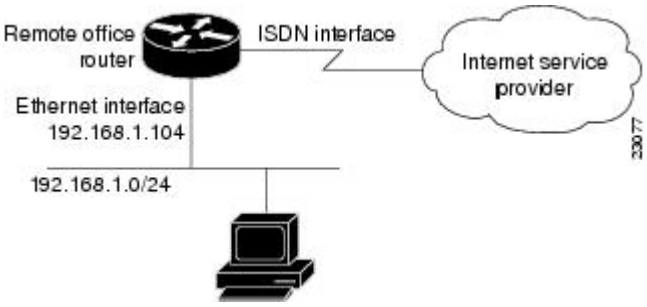
Note For Frame Relay or ATM interfaces, you can apply CBAC inspection rules separately on each sub-interface, even though the subinterfaces are physically connected through one interface.

```
! -------------------------
! 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 5 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
```

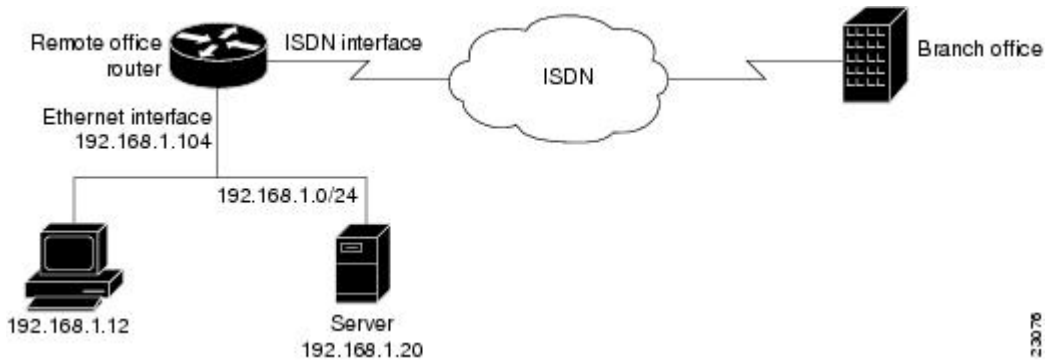
```
! interface where this rule is applied.
!
! Deny broadcast messages with a source address of 255.255.255.255; this helps to
! prevent broadcast attacks.
access-list 105 deny ip host 255.255.255.255 any
!
! Add anti-spoofing protection by denying traffic with a source address matching a host
! on 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 6 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
```

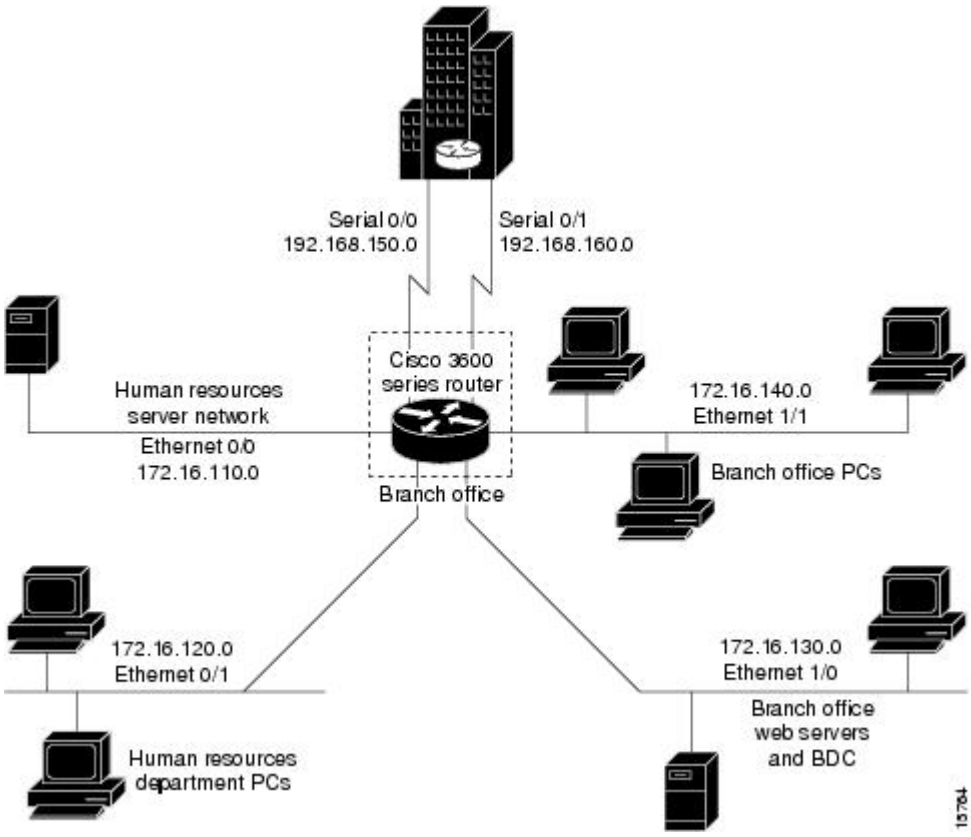
```
exec-timeout 0 0
password 7 <elided>
login local
line vty 2
exec-timeout 0 0
password 7 <elided>
login local
line vty 3
exec-timeout 0 0
password 7 <elided>
login local
line vty 4
exec-timeout 0 0
password 7 <elided>
login local
!
scheduler interval 500
end
```

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 7 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.
© 2007 Cisco Systems, Inc. All rights reserved.
```

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 feature information and caveats, see 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 document.

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.

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.

Table 5 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)
• 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 6 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.
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.

Connection Type	Description
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.
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 7 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.

Connection Type	Description
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.
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 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.

	Command or Action	Purpose
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.
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, page 78

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** **Example:** `Router> enable`	Enables privileged EXEC mode. • 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, page 80

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 8 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**.

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 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 feature information and caveats, see 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 document.

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, page 91
- Verifying CBAC Configurations, page 92

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 9 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**.

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 feature information and caveats, see 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 document.

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

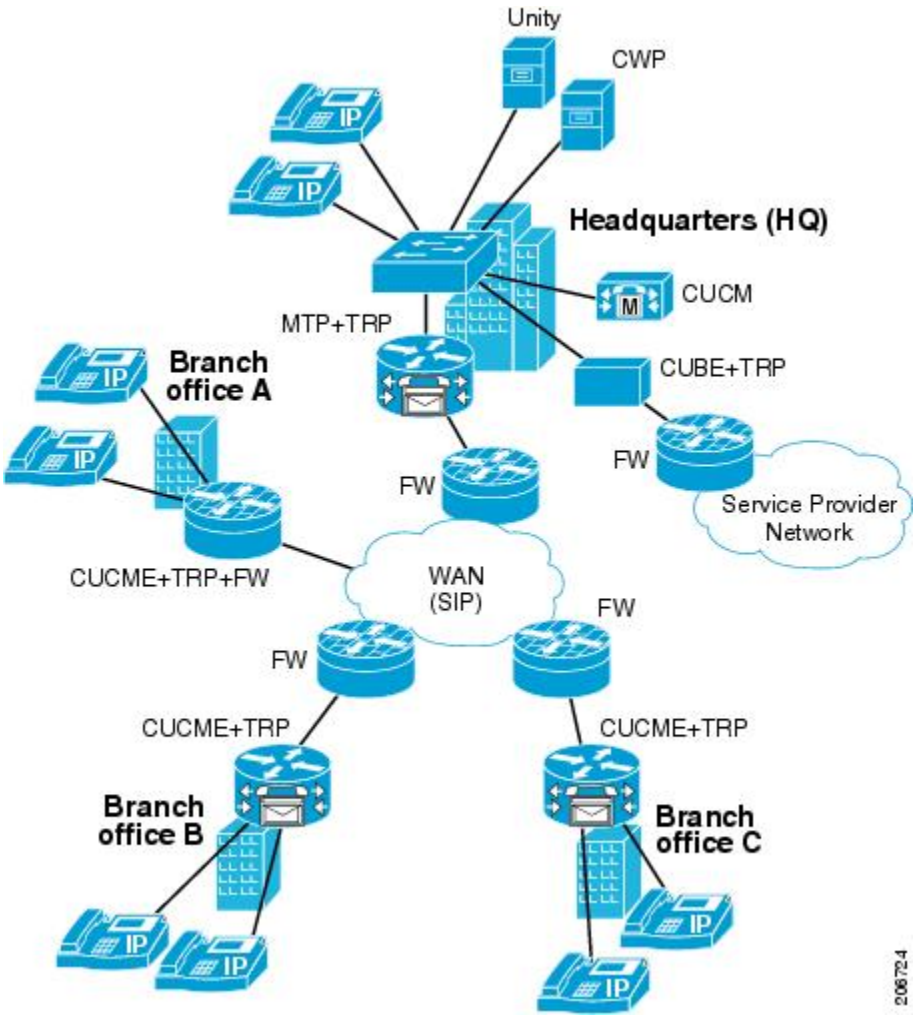
The Cisco IOS Firewall extends the concept of static access control lists (ACLs) by introducing dynamic ACL entries that open on the basis of the necessary application ports on a specific application and close these ports at the end of the application session. The Cisco IOS Firewall achieves this functionality 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.

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.
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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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.

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.
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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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, page 105

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 10 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**.

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 feature information and caveats, see 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 document.

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, page 110
- Example New show ip inspect CLI Output, page 111

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 IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	*Cisco IOS Security Command Reference*

Standards

Standards	Title
None	--

MIBs

MIBs	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

RFCs	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 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 11 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.

Firewall Websense URL Filtering

The Firewall Websense URL Filtering feature enables your Cisco IOS firewall (also known as Cisco Secure Integrated Software [CSIS]) to interact with the Websense URL filtering software, thereby allowing you to prevent users from accessing specified websites on the basis of some policy. The Cisco IOS firewall works with the Websense server to know 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 feature information and caveats, see 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 document.

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 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 look-up requests will be sent only to the primary server.

URL Filtering Support Restriction

This feature supports only one active URL filtering scheme at a time. (Before enabling Websense URL filtering, you should always ensure that there is not another URL filtering scheme configured, such as N2H2.)

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 another mechanism for getting the username to correspond to an IP address.

Exclusive Domain List Restriction

This feature does not resolve the domains before it searches an exclusive domain list. When a questionable URL is presented to the filtering server, this feature searches only for the value that was specified in the command-line interface (CLI). That is, if an exclusive domain list was configured via the **ip urlfilter exclusive-domain deny 198.168.1.1**command, a user entering http://198.168.1.1. into a browser will be denied access. However, a user who is trying to access this same domain and who enters http://www.cisco.com, will be allowed access because 198.168.1.1 was specified via the CLI, not www.cisco.com.

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

- Only one inspection rule is supported.
- Only HTTP filtering is supported. (HTTPS and FTP filtering are not supported.)
- HTTP over ports used by static Network Based Application Recognition (NBAR) protocols are not supported.
- Context-based Access Control (CBAC) is not supported.
- Only Layer 3 SVIs, Layer 3 routed ports, and Layer 3 subinterfaces are supported.
- The **clear ip urlfilter cache** and **show ip urlfilter cache** commands are not supported.
- Only the Websense URL filtering server is supported. (N2H2/SmartFilter/Trend Micro filtering servers are not supported.)
- Usernames are not passed on from PISA to Websense.

Information About Firewall Websense URL Filtering

Benefits of Firewall Websense URL Filtering

The Cisco IOS Firewall Websense 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. In addition, the following functions are available in this feature:

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.

A firewall marks a primary server as down when sending a request to or receiving a response from the server fails. When a primary server goes down, the system will go to the beginning of the configured servers list and try to activate the first server on the list. If the first server on the list is unavailable, it will try the second server on the list; the system will keep 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, it will set a flag indicating that all of the servers are down, and it will enter allow mode.

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

IP Cache Table

This function provides an IP cache table that contains the 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. If the cached IP addresses have exceeded 80 percent, it will start removing idle entries; if it has not exceeded 80 percent, it will quit and wait for the next cycle. The absolute timer is a large periodic timer (1 hour) that is used to remove all of the elapsed entries. (The age of an elapsed entry is greater than the absolute time.) An elapsed entry will also be removed during cache lookup.

The idle time value is fixed at 10 minutes. The absolute time value is taken from the Websense look-up response, which is often greater than 15 hours. The absolute value for cache entry made out 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.

Packet Buffering

This function allows you to increase the maximum number of HTTP responses that a Cisco IOS firewall can hold. If the HTTP responses arrive prior to a Websense server reply, this buffering scheme allows your firewall to store a maximum of 200 HTTP responses. (After 200 responses have been reached, the firewall will drop further responses.) The responses will remain in the buffer until an allow or deny message is received from Websense: if the status indicates that the URL is allowed, the firewall will release the HTTP responses in the buffer to the browser of the end user; if the status indicates that the URL is blocked, the firewall will discard the HTTP responses in the buffer and close the connection to both ends. This function prevents numerous HTTP responses from overwhelming your system.

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

Exclusive Domains

This function provides a configurable list of domain names so that the Cisco IOS 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 have to deal with look-up 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, the user 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 only a partial domain name to the exclusive domain list, such as ".cisco.com," 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

The system will go into allow mode when connections to all the Websense servers are down. The system will return to normal mode when a connection to at least one web 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.

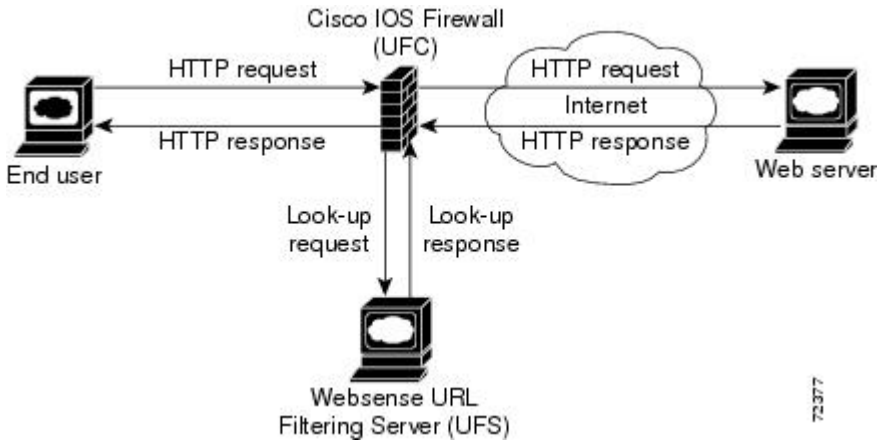
To configure allow mode for your system, use the **ip urlfilter allowmode** command.

Feature Design of Firewall Websense URL Filtering

Note This feature assumes that the Websense server will be part of a protected network and that requests from the Cisco IOS 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 Cisco IOS firewall receives this request, it forwards the request to the web server while simultaneously extracting the URL and sending a look-up request to the Websense server.
3. After the Websense server receives the look-up request, it checks its database to see whether it should permit or deny the URL; it returns a permit or deny status via a look-up response to the Cisco IOS firewall.
4. After the Cisco IOS firewall receives this look-up response, it performs one of the following functions:
5. If the look-up response permits the URL, it sends the HTTP response to the end user.

6 If the look-up 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 to both ends.

Supported Websense Server Features on a Cisco IOS Firewall

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

The following filtering methods are supported:

- Global filtering, which is applied to all users, groups, and IP addresses
- User- or group-based filtering, which is applied to a specific user or group
- Keyword-based filtering, which is applied on the basis of specific keywords (for example, a user can configure a policy for which all URLs with the keyword “dog” will be denied)
- Category-based filtering, which is applied on the basis of specific categories
- Customized filtering, which allows the user to apply a policy for customized URLs

The NT LAN Manager (NTLM) and Lightweight Directory Access Protocol (LDAP) user authentication methods are supported in this feature. Websense uses these methods to authenticate the user when the firewall does not pass the authenticated username along with the look-up request.

When the username is not passed along with the look-up request, the Websense server retrieves the username through one of the following methods:

- Manual authentication--Websense 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, Websense checks the NTLM or LDAP directory service to see if the username and password are a match. If there is a match, Websense associates the username with the source IP address and policies can be created for that username.
- Transparent ID (XID)--Websense has an agent that automatically associates users, when they log onto a Windows network, to their IP addresses. Unlike manual authentication, this method does not require an additional logon by the user. However, this method can be used only for Windows.

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

How to Configure Websense URL Filtering

Configuring Firewall Websense URL Filtering

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

Before enabling Websense URL filtering, you should always ensure that there is not another URL filtering scheme 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 co-exist.”

Note Enabling HTTP inspection (via 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 and argument and configure a standard access list to allow any traffic. Configuring URL filtering without enabling the **java-list** *access-list*keyword and argument 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**

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* http [java-list access-list] [urlfilter] [alert {on | off}] [audit-trail {on | off}] [timeout seconds] **Example:** **Example:** `Router(config)# ip inspect name fw_urlf http java-list 51 urlfilter timeout 30`	Turns on HTTP inspection. The **urlfilter** keyword associates URL filtering with HTTP inspection. **Note** You may configure two or more inspections in a router, but URL filtering will work only with the inspections in which the **urlfilter** keyword is enabled. **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 and argument. Configuring URL filtering without enabling the **java-list** *access-list*keyword and argument will severely impact performance.
Step 4	**ip inspect** *inspection-name* {**in** | **out**} **Example:** `Router(config)# ip inspect fw_urlf in`	Applies a set of inspection rules to an interface. • The **in** keyword applies the inspection rules to inbound traffic.
Step 5	**ip urlfilter server vendor websense n2h2** } *ip-address* [**port** *port-number*] [**timeout** *seconds*] [**retransmit** *number*] **Example:** `Router(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. • *ip-address* --IP address of the vendor server. • **port** *port-number* --Port number that the vendor server listens on. The default port number is 15868. • **timeout** *seconds* --Length of time that the firewall will wait for a response from the vendor server. The default timeout is 5 minutes. • **retransmit** *number* --Number of times the firewall will retransmit the request when a response does not arrive. The default value is 2 times.

	Command or Action	Purpose
Step 6	**ip urlfilter alert** **Example:** `Router(config)# ip urlfilter alert`	(Optional) Enables the system alert, which displays system messages such as a server entering allow mode or going down. • The system alert is enabled by default.
Step 7	**ip urlfilter audit-trail** **Example:** `Router(config)# ip urlfilter audit-trail`	(Optional) Enables the logging of messages into the syslog server of router. This function is disabled by default.
Step 8	**ip urlfilter urlf-server-log** **Example:** `Router(config)# ip urlfilter urlf-server-log`	(Optional) Enables the logging of system messages on the URL filtering server (the Websense server). • This function is disabled by default.
Step 9	**ip urlfilter exclusive-domain permit** | **deny**} *domain-name* **Example:** `Router(config)# ip urlfilter exclusive-domain permit www.cisco.com`	(Optional) Adds a domain name to or from the exclusive domain list so that the firewall does not have to send look-up requests to the Websense server. • **permit** --Permits all traffic destined for the specified domain name. • **deny** --Denies all traffic destined for the specified domain name. • *domain-name* --Domain name that is added or removed from the exclusive domain list.
Step 10	**ip urlfilter cache** *number* **Example:** `Router(config)# ip urlfilter cache 4500`	(Optional) Configures cache table parameters. • *number* --Maximum number of destination IP addresses that can be cached into the cache table; the default is 5000.
Step 11	**ip urlfilter allowmode** [**on** | **off**] **Example:** `Router(config)# ip urlfilter allowmode on`	(Optional) Turns on the default mode of the filtering systems. • **on** --Allows HTTP requests to pass to the end user if all Websense servers are down. • **off** --Blocks all HTTP requests if all Websense servers are down; **off** is the default setting.

	Command or Action	Purpose
Step 12	**ip urlfilter max-resp-pak** *number* **Example:** `Router(config)# ip urlfilter max-resp-pak 150`	(Optional) Configures the maximum number of HTTP responses that the firewall can keep in its packet buffer. The default value is 200 512-byte buffers.
Step 13	**ip urlfilter max-request** *number* **Example:** `Router(config)# ip urlfilter maxrequest 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. • The default value is 1000.
Step 14	**ip urlfilter truncate** {**script-parameters** | **hostname**} **Example:** `Router(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:** `Router(config)# ip urlfilter mode per-uri`	(Optional) Configures a URL filtering mode. • **per-session** --Filters the first URL in the HTTP session. • **per-uri** --Filters the first URL in each packet. • **per-uri-proxy-only** --Filters via the **per-session** keyword behavior for direct (non-proxy) requests. Filters via the **per-uri**keyword behavior for proxy requests. **Note** This command is available only on the Catalyst 6500 with PISA in Cisco IOS Release 12.2(18)ZYA.

• Troubleshooting Tips, page 123

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 level three LOG_ERR-type message is displayed when a configured UFS goes down. When this happens, the firewall will mark the configured server as secondary and try to bring up one of the other secondary servers and mark that server as the primary server. If there is no other server configured, the firewall will enter allow mode and display 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 will be triggered that will try to bring up a server by opening a TCP connection.

- "%URLF-5-SERVER_UP: Connection to an URL filter server 10.92.0.9 is made; the system is returning from ALLOW MODE"

This LOG_NOTICE-type message is displayed when the 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 look-up request is too long; any URL longer than 3K will be dropped.

- "%URLF-4-MAX_REQ: The number of pending request 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 12.54.192.6:54678 server 64.192.14.2:80"

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 will be truncated to 300 bytes and then logged.

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

Verifying Cisco IOS Firewall and Websense URL Filtering

To verify that the Firewall Websense URL Filtering feature is working, perform any of the following optional steps:

Command or Action	Purpose
enable `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

show ip urlfilter cache
`Router#`
`show ip urlfilter cache`
	Displays the destination IP addresses that are cached into the cache table.
	Note This command is not supported on PISA in Cisco IOS Release 12.2(18)ZYA.

show ip urlfilter config
`Router#`
`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 `Router# 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, the number of blocked URLs.

Maintaining the Cache Table

To clear the cache table of a specified address or of all IP addresses, perform the following optional steps.

Command or Action	Purpose
enable `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

Command or Action	Purpose
clear ip urlfilter cache {*ip-address* | **all**	Clears the cache table. **Note** This command is not supported on PISA in Cisco IOS Release 12.2(18)ZYA.

```
Router#
clear ip urlfilter cache all
```

Monitoring the URL Filter Subsystems

To monitor the URL filter subsystems, perform the following optional steps:

Command or Action	Purpose
enable	Enables privileged EXEC mode. • Enter your password if prompted.

```
Router> enable
```

Command or Action	Purpose
debug ip urlfilter {**func-trace** | **detailed** | **events**}	Enables debugging information of the URL filter subsystems. • **func-trace** --Prints a sequence of important functions that are called when configuring URL filtering. • **detailed** --Prints detailed information about various activities that occur during URL filtering. • **events** --Prints various events, such as queue event, timer event, and socket event.

```
Router#
debug ip urlfilter detailed
```

Configuration Examples for the Firewall and Webserver

Example URL Filter Client (Firewall) Configuration

The following example shows how to configure the Cisco IOS firewall (also known as the URL filter client [UFC]) for 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 .cat.com
ip urlfilter exclusive-domain deny .dog.com
ip urlfilter exclusive-domain permit www.store.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 FastEthernet0/0
 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 Ethernet1/0
 ip address 10.6.9.7 255.255.0.0
 ip nat outside
 no ip route-cache
 no ip mroute-cache
 duplex half
!
interface Ethernet1/1
 no ip address
 no ip mroute-cache
 shutdown
 duplex half
!
interface Ethernet1/2
 no ip address
 no ip mroute-cache
 shutdown
 duplex half
!
interface Ethernet1/3
 no ip address
 no ip mroute-cache
 shutdown
 duplex half
!
interface Serial2/0
 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 sisu-devtest/coredump/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	Cisco IOS Master Commands List, All Releases
Additional firewall commands	*Cisco IOS Security Command Reference*
N2H2 URL filtering	The chapter "Firewall N2H2 Support", in the *Cisco IOS Security Configuration Guide*

Standards

Standards	Title
None	--

MIBs

MIBs	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

RFCs[6]	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.

6 Not all supported RFCs are listed.

Table 12 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	This feature enables your Cisco IOS 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 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 server vendor, ip urlfilter urlf-server-log, show ip urlfilter cache**, **show ip urlfilter config**, **show ip urlfilter statistics**. In Cisco IOS Release 12.2(18)ZYA, the following command was introduced: **ip urlfilter mode**

Glossary

CSIS--Cisco Secure Integrated Software. CSIS is a content-based firewall that currently inspects application data, checks for protocol conformance, extracts the relevant port information to create dynamic access list entries that successfully allow return traffic, and closes the ports at the end of the session.

UFC--URL filter client. UFC is a separate process that accepts URLs from CSIS, forwards the URL to the Websense server, and processes the 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.

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

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 feature information and caveats, see 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 document.

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, page 134
- Cisco IOS HTTP Application Policy Overview, page 134

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, page 134
- Applying an HTTP Application Policy to a Firewall for Inspection, page 139

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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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] **Example:** `Router(cfg-appfw-policy-http)# request-method rfc default action allow alarm`	(Optional) Permits or denies HTTP traffic according to either the request methods or the extension methods. • **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.

Command or Action	Purpose
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.
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, page 139

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).

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.

Command or Action	Purpose
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.
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.`

Command or Action	Purpose

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, page 141

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 13 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**.

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 feature information and caveats, see 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 document.

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.

- The Cisco IOS Firewall supports only Version 2 of the H.323 protocol. If CCME or the H.323 Gateway has inspection of H.323 router traffic enabled, enter the following commands so that it is configured 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 14 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.

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 feature information and caveats, see 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 document.

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.

- 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.
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.

	Command or Action	Purpose
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, page 163
- What to Do Next, page 163

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.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
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**
 	
Example:
`Router(config-if)`
`ip access-group 156 in`
	Controls access to an interface.

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** **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	**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** **Example:** `Router# debug ip inspect L2-transparent dhcp-passthrough`	Enables debugging messages for transparent firewall events. • **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.

	Command or Action	Purpose

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:

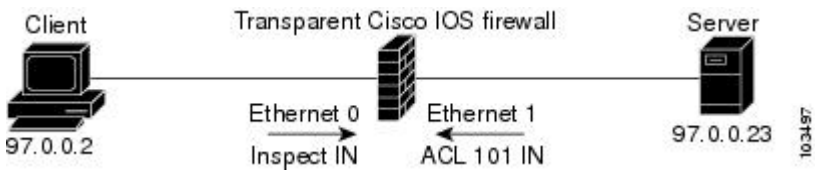
```
Router# show ip inspect all
.
.
.
Interface Configuration
! Below is the bridging interface.
Interface Ethernet1
Inbound inspection rule is test
tcp alert is on audit-trail is off timeout 3600
ftp alert is on audit-trail is off timeout 3600
Outgoing inspection rule is not set
Inbound access list 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, page 174
- Example Denying DHCP Pass-Through Traffic, page 174

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 15 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.

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.

Virtual Fragmentation Reassembly

Currently, the Cisco IOS Firewall--specifically context-based access control (CBAC) and the intrusion detection system (IDS)--cannot identify the contents of the IP fragments nor can it gather port information from the fragment. These inabilities allow the fragments to pass through the network without being examined or without dynamic access control list (ACL) creation.

Virtual fragmentation reassembly (VFR) enables the Cisco IOS Firewall to create the appropriate dynamic ACLs, thereby, protecting the network from various fragmentation attacks.

Feature History for Virtual Fragmentation Reassembly

Release	Modification
12.3(8)T	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

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

Restrictions for Virtual Fragmentation Reassembly

Performance Impact

VFR will cause a performance impact on the basis of functions such as packet copying, fragment validation, and fragment reorder. This performance impact will vary depending on the number of concurrent IP datagram that are being reassembled.

VFR Configuration Restriction

VFR should not be enabled on a router that is placed on an asymmetric path. The reassembly process requires all of the fragments within an IP datagram. Routers placed in the asymmetric path may not receive all of the fragments, so the fragment reassembly will fail.

SIP and RTSP Limitation

The Session Initiation Protocol (SIP) and the Real-Time Streaming Protocol (RTSP) do not have the ability to parse port information across noncontiguous buffers. Thus, virtual fragmentation reassembly may fail. (If the application fails, the session will be blocked.)

Information About Virtual Fragmentation Reassembly

To use fragmentation support for Cisco IOS Firewall, you should understand the following concept:

- Detected Fragment Attacks, page 180
- Automatically Enabling or Disabling VFR, page 181

Detected Fragment Attacks

VFR is responsible for detecting and preventing the following types of fragment attacks:

- Tiny Fragment Attack--In this type of attack, the attacker makes the fragment size small enough to force Layer 4 (TCP and User Datagram Protocol (UDP)) header fields into the second fragment. Thus, the ACL rules that have been configured for those fields will not match.

VFR drops all tiny fragments, and an alert message such as follows is logged to the syslog server: “VFR-3-TINY_FRAGMENTS.”

- Overlapping Fragment Attack--In this type of attack, the attacker can overwrite the fragment offset in the noninitial IP fragment packets. When the firewall reassembles the IP fragments, it might create wrong IP packets, causing the memory to overflow or your system to crash.

VFR drops all fragments within a fragment chain if an overlap fragment is detected, and an alert message such as follows is logged to the syslog server: “VFR-3-OVERLAP_FRAGMENT.”

- Buffer Overflow Attack--In this type of denial-of-service (DoS) attack, the attacker can continuously send a large number of incomplete IP fragments, causing the firewall to lose time and memory while trying to reassemble the fake packets.

To avoid buffer overflow and control memory usage, configure a maximum threshold for the number of IP datagrams that are being reassembled and the number of fragments per datagram. (Both of these parameters can be specified via the **ip virtual-reassembly** command.)

When the maximum number of datagrams that can be reassembled at any given time is reached, all subsequent fragments are dropped, and an alert message such as the following is logged to the syslog server: “VFR-4_FRAG_TABLE_OVERFLOW.”

When the maximum number of fragments per datagram is reached, subsequent fragments will be dropped, and an alert message such as the following is logged to the syslog server: “VFR-4_TOO_MANY_FRAGMENTS.”

In addition to configuring the maximum threshold values, each IP datagram is associated with a managed timer. If the IP datagram does not receive all of the fragments within the specified time, the timer will expire and the IP datagram (and all of its fragments) will be dropped.

Automatically Enabling or Disabling VFR

VFR is designed to work with any feature that requires fragment reassembly (such as Cisco IOS Firewall and NAT). Currently, NAT enables and disables VFR internally; that is, when NAT is enabled on an interface, VFR is automatically enabled on that interface.

If more than one feature attempts to automatically enable VFR on an interface, VFR will maintain a reference count to keep track of the number of features that have enabled VFR. When the reference count is reduced to zero, VFR is automatically disabled.

How to Use Virtual Fragmentation Reassembly

Configuring VFR

Use this task to enable VFR on an interface, specify maximum threshold values to combat buffer overflow and control memory usage, and verify any VFR configurations.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. ip virtual-reassembly [max-reassemblies number] [max-fragments number] [timeout seconds] [drop-fragments]
5. **exit**
6. **exit**
7. show ip virtual-reassembly [interface type]

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	**interface** *type number* **Example:** `Router(config)# interface ethernet1/1`	Configures an interface type and enters interface configuration mode.
Step 4	ip virtual-reassembly [max-reassemblies number] [max-fragments number] [timeout seconds] [drop-fragments] **Example:** `Router(config-if)# ip virtual-reassembly max-reassemblies 64 max-fragments 16 timeout 5`	Enables VFR on an interface.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	show ip virtual-reassembly [interface type] **Example:** `Router# show ip virtual-reassembly ethernet1/1`	Displays the configuration and statistical information of the VFR. If an interface is not specified, VFR information is shown for all configured interfaces.

- Troubleshooting Tips, page 182

Troubleshooting Tips

To view debugging messages related to the VFR subsystem, use the **debug ip virtual-reassembly** command.

Configuration Examples for Fragmentation Reassembly

Additional References

The following sections provide references related to virtual fragmentation reassembly.

Related Documents

Related Topic	Document Title
Dynamic IDS	*Cisco IOS Intrusion Prevention System*
CBAC	*Configuring Context-Based Access Control*

Standards

Standards	Title
None	--

MIBs

MIBs	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

RFCs	Title
RFC 791	Internet Protocol
RFC 1858	Security Considerations for IP Fragment Filtering

Technical Assistance

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

Command Reference

The following commands are introduced or modified in the feature or features documented in this module. For information about these commands, see the *Cisco IOS Security Command Reference*. For information

about all Cisco IOS commands, go to the Command Lookup Tool at http://tools.cisco.com/Support/CLILookup or to the *Cisco IOS Master Commands List*.

- **debug ip virtual-reassembly**
- **ip virtual-reassembly**
- **show ip virtual-reassembly**

Glossary

fragment --Part of an IP datagram that is fragmented into multiple pieces. Each piece is called a fragment or an IP fragment.

fragmentation --Process of breaking down an IP datagram into smaller packets (fragments) that are transmitted over different types of network media.

initial fragment -- First fragment within a fragment set. This fragment should have a Layer 4 header and should have an offset of zero.

noninitial fragment --All fragments within a fragment set, except the initial fragment.

VRF Aware Cisco IOS Firewall

VRF Aware Cisco IOS Firewall applies Cisco IOS Firewall functionality to VRF (Virtual Routing and Forwarding) interfaces when the firewall is configured on a service provider (SP) or large enterprise edge router. SPs can provide managed services to small and medium business markets.

The VRF Aware Cisco IOS Firewall supports VRF-aware URL filtering and VRF-lite (also known as Multi-VRF CE).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see 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 document.

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 VRF Aware Cisco IOS Firewall

- Understand Cisco IOS firewalls.
- Configure VRFs.
- Verify that the VRFs are operational.

Restrictions for VRF Aware Cisco IOS Firewall

- VRF Aware Cisco IOS Firewall is not supported on Multiprotocol Label Switching (MPLS) interfaces.

- If two VPN networks have overlapping addresses, VRF-aware network address translation (NAT) is required for them to support VRF-aware Firewalls.
- When crypto tunnels belonging to multiple VPNs terminate on a single interface, you cannot apply per-VRF firewall policies.

Information About VRF Aware Cisco IOS Firewall

Cisco IOS Firewall

The Cisco IOS Firewall provides robust, integrated firewall and intrusion detection functionality for every perimeter of the network. Available for a wide range of Cisco IOS software-based routers, the Cisco IOS Firewall offers sophisticated security and policy enforcement for connections within an organization (intranet) and between partner networks (extranets), as well as for securing Internet connectivity for remote and branch offices.

The Cisco IOS Firewall enhances existing Cisco IOS security capabilities such as authentication, encryption, and failover, with state-of-the-art security features such as stateful, application-based filtering (context-based access control), defense against network attacks, per-user authentication and authorization, and real-time alerts.

The Cisco IOS Firewall is configurable via Cisco ConfigMaker software, an easy-to-use Microsoft Windows 95, Windows 98, NT 4.0 based software tool.

The Cisco IOS Firewall provides great value in addition to these benefits:

- Flexibility--Provides multiprotocol routing, perimeter security, intrusion detection, VPN functionality, and dynamic per-user authentication and authorization.
- Scalable deployment--Scales to meet any network's bandwidth and performance requirements.
- Investment protection--Leverages existing multiprotocol router investment.
- VPN support--Provides a complete VPN solution based on Cisco IOS IPSec and other CISCO IOS software-based technologies, including L2TP tunneling and quality of service (QoS).

The VRF Aware Cisco IOS Firewall is different from the non-VRF Aware Firewall because it does the following:

- Allows users to configure a per-VRF Firewall. The firewall inspects IP packets that are sent and received within a VRF.
- Allows SPs to deploy the firewall on the provider edge (PE) router.
- Supports overlapping IP address space, thereby allowing traffic from nonintersecting VRFs to have the same IP address.

- Supports per-VRF (not global) firewall command parameters and Denial-of-Service (DoS) parameters so that the VRF-aware Firewall can run as multiple instances (with VRF instances) allocated to various Virtual Private Network (VPN) customers.
- Performs per-VRF URL filtering.
- Generates VRF-specific syslog messages that can be seen only by a particular VPN. These alert and audit-trail messages allow network administrators to manage the firewall; that is, they can adjust firewall parameters, detect malicious sources and attacks, add security policies, and so forth. The vrf name is tagged to syslog messages being logged to the syslog server.

Both VFR Aware and non-VFR Aware Firewalls now allow you to limit the number of firewall sessions. Otherwise, it would be difficult for VRFs to share router resources because one VRF may consume a maximum amount of resources, leaving few resources for other VRFs. That would cause the denial of service to other VRFs. To limit the number of sessions, enter the **ipinspectname** command.

VRF

VPN Routing and Forwarding (VRF) is an IOS route table instance for connecting a set of sites to a VPN service. A VRF contains a template of a VPN Routing/Forwarding table in a PE router.

The overlapping addresses, usually resulting from the use of private IP addresses in customer networks, are one of the major obstacles to successful deployment of peer-to-peer VPN implementation. The MPLS VPN technology provides a solution to this dilemma.

Each VPN has its own routing and forwarding table in the router, so any customer or site that belongs to a VPN is provided access only to the set of routes contained within that table. Any PE router in the MPLS VPN network therefore contains a number of per-VPN routing tables and a global routing table that is used to reach other routers in the service provider network. Effectively, a number of virtual routers are created in a single physical router.

VRF-lite

VRF-lite is a feature that enables a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. VRF-lite uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF. Interfaces in a VRF can be physical, such as Ethernet ports, or logical, such as VLAN switched virtual interfaces (SVIs). However, a Layer 3 interface cannot belong to more than one VRF at a time.

Note VRF-lite interfaces must be Layer 3 interfaces.

VRF-lite includes these devices:

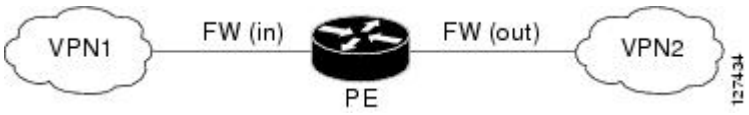
- Customer edge (CE) devices provide customer access to the service provider network over a data link to one or more provider edge (PE) routers. The CE device advertises the site's local routes to the PE router and learns the remote VPN routes from it. A Catalyst 4500 switch can be a CE.
- Provider edge (PE) routers exchange routing information with CE devices by using static routing or a routing protocol such as BGP, RIPv1, or RIPv2.
- Provider routers (or core routers) are any routers in the service provider network that do not attach to CE devices.
- The PE is only required to maintain VPN routes for those VPNs to which it is directly attached, eliminating the need for the PE to maintain all of the service provider VPN routes. Each PE router maintains a VRF for each of its directly connected sites. Multiple interfaces on a PE router can be associated with a single VRF if all of these sites participate in the same VPN. Each VPN is mapped to

a specified VRF. After learning local VPN routes from CEs, a PE router exchanges VPN routing information with other PE routers by using internal BGP (IBPG).

With VRF-lite, multiple customers can share one CE, and only one physical link is used between the CE and the PE. The shared CE maintains separate VRF tables for each customer, and switches or routes packets for each customer based on its own routing table. VRF-lite extends limited PE functionality to a CE device, giving it the ability to maintain separate VRF tables to extend the privacy and security of a VPN to the branch office.

In a VRF-to-VRF situation, if firewall policies are applied on both inbound and outbound interfaces as shown in the figure below, the firewall on the inbound interface takes precedence over the firewall on the outbound interface. If the incoming packets do not match against the firewall rules (that is, the inspection protocols) configured on the inbound interface, the firewall rule on the outbound interface is applied to the packet.

Figure 11 Firewall in a VRF-to-VRF Scenario



Per-VRF URL Filtering

The VRF-aware firewall supports per-VRF URL filtering. Each VPN can have its own URL filter server. The URL filter server typically is placed in the shared service segment of the corresponding VPN. (Each VPN has a VLAN segment in the shared service network.) The URL filter server can also be placed at the customer site.

AlertsandAuditTrails

CBAC generates real-time alerts and audit trails based on events tracked by the firewall. Enhanced audit trail features use SYSLOG to track all network transactions; recording time stamps, the source host, the 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.

MPLS VPN

The MPLS VPN feature allows multiple sites to interconnect transparently through a service provider network. One service provider network can support several IP VPNs. Each VPN appears to its users as a private network, separate from all other networks. Within a VPN, each site can send IP packets to any other site in the same VPN.

Each VPN is associated with one or more VPN VRF instances. A VRF consists of an IP routing table, a derived Cisco Express Forwarding table, and a set of interfaces that use the forwarding table.

The router maintains a separate routing and Cisco Express Forwarding tables for each VRF. This prevents information from being sent outside the VPN and allows the same subnet to be used in several VPNs without causing duplicate IP address problems.

The router using Multiprotocol BGP (MP-BGP) distributes the VPN routing information using the MP-BGP extended communities.

VRF-aware NAT

Network Address Translation (NAT) allows a single device, such as a router, to act as an agent between the Internet (or public network) and a local (or private) network. Although NAT systems can provide broad levels of security advantages, their main objective is to economize on address space.

NAT allows organizations to resolve the problem of IP address depletion when they have existing networks and need to access the Internet. Sites that do not yet possess NIC-registered IP addresses must acquire them. Cisco IOS NAT eliminates concern and bureaucratic delay by dynamically mapping thousands of hidden internal addresses to a range of easy-to-get addresses.

In general, a NAT system makes it more difficult for an attacker to determine the following:

- Number of systems running on a network
- Type of machines and operating systems they are running
- Network topology and arrangement

NAT integration with MPLS VPNs allows multiple MPLS VPNs to be configured on a single device to work together. NAT can differentiate which MPLS VPN it receives IP traffic from even if the MPLS VPNS are all using the same IP addressing scheme. This enables multiple MPLS VPN customers to share services while ensuring that each MPLS VPN is completely separate from the other.

MPLS service providers would like to provide value-added services such as Internet connectivity, domain name servers (DNS), and VoIP service to their customers. This requires that their customers IP addresses be different when reaching the services. Because MPLS VPN allows customers to use overlapped IP addresses in their networks, NAT must be implemented to make the services possible.

There are two approaches to implementing NAT in the MPLS VPN network. NAT can be implemented on the CE router, which is already supported by NAT, or it can be implemented on a PE router. The NAT Integration with MPLS VPNs feature enables the implementation of NAT on a PE router in an MPLS cloud.

VRF-aware IPSec

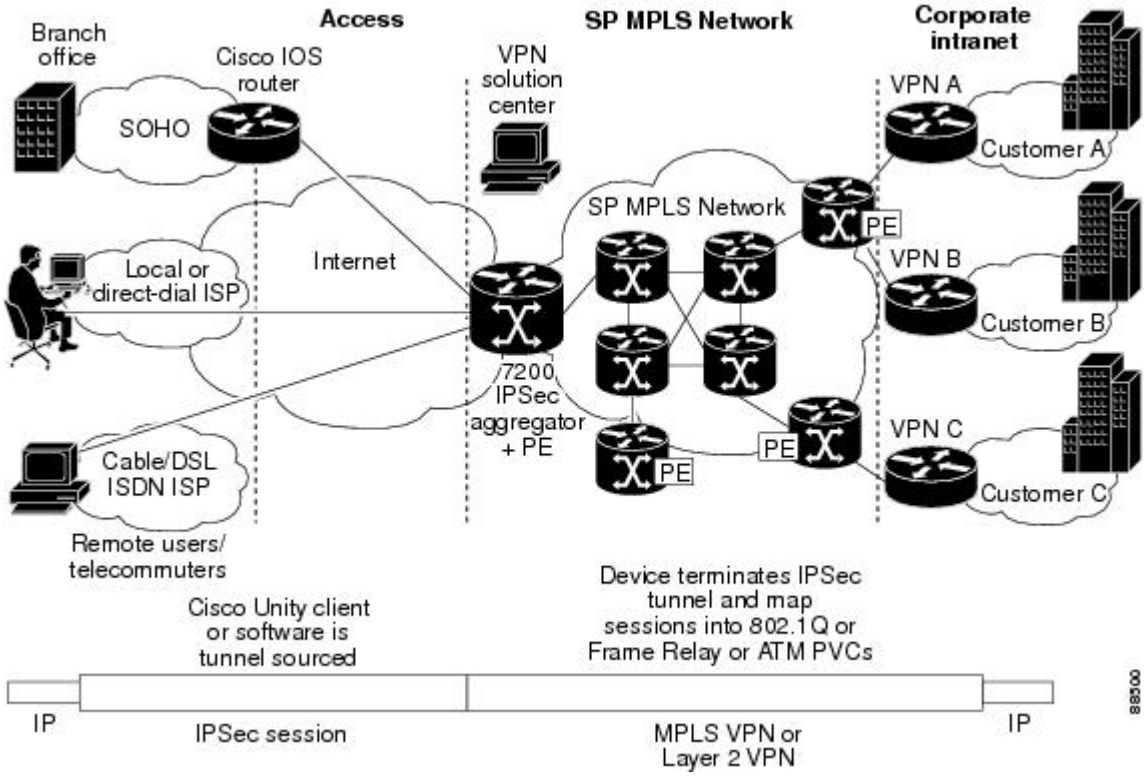
The VRF-aware IPSec feature maps an IP Security (IPSec) tunnel to an MPLS VPN. Using the VRF-aware IPSec feature, you can map IPSec tunnels to VRF instances using a single public-facing address.

Each IPSec tunnel is associated with two VRF domains. The outer encapsulated packet belongs to a VRF domain called the Front Door VRF (FVRF). The inner, protected IP packet belongs to a domain called the Inside VRF (IVRF). In other words, the local endpoint of the IPSec tunnel belongs to the FVRF, whereas the source and destination addresses of the inside packet belong to the IVRF.

One or more IPSec tunnels can terminate on a single interface. The FVRF of all these tunnels is the same and is set to the VRF that is configured on that interface. The IVRF of these tunnels can be different and depends on the VRF that is defined in the Internet Security Association and Key Management Protocol (ISAKMP) profile that is attached to a crypto map entry.

The figure below illustrates a scenario showing IPSec to MPLS and Layer 2 VPNs.

Figure 12 IPSec-to-MPLS and Layer 2 VPNs



VRF Aware Cisco IOS Firewall Deployment

A firewall can be deployed at many points within the network to protect VPN sites from Shared Service (or the Internet) and vice versa. The following firewall deployments are described:

- Distributed Network Inclusion of VRF Aware Cisco IOS Firewall, page 190
- Hub-and-Spoke Network Inclusion of VRF Aware Cisco IOS Firewall, page 192

Distributed Network Inclusion of VRF Aware Cisco IOS Firewall

A VRF Aware Cisco IOS Firewall in a distributed network has the following advantages:

- The firewall is distributed across the MPLS core, so the firewall processing load is distributed to all ingress PE routers.
- VPN Firewall features can be deployed in the inbound direction.
- Shared Service is protected from the VPN site at the ingress PE router; therefore, malicious packets from VPN sites are filtered at the ingress PE router before they enter the MPLS core.

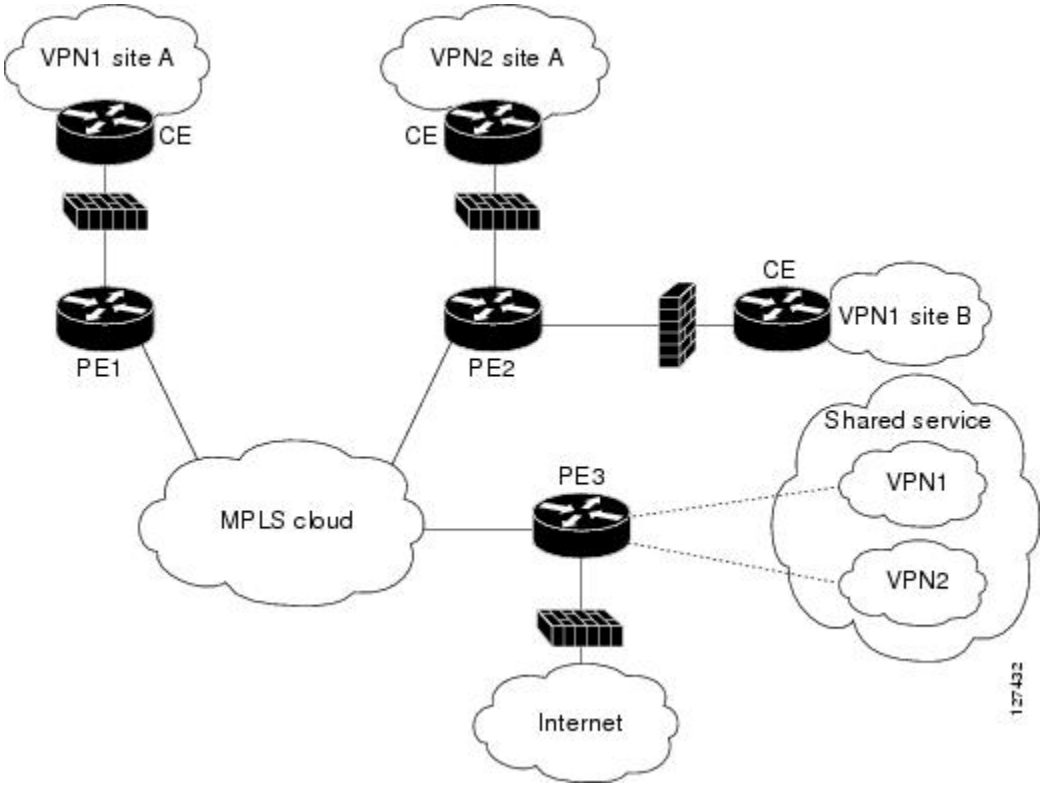
However, the following disadvantages exist:

- There is no centralized firewall deployment, which complicates the deployment and management of the firewall.
- Shared Service firewall features cannot be deployed in the inbound direction.

- The MPLS core is open to the Shared Service. Therefore, malicious packets from Shared Service are filtered only at the ingress PE router after traveling through all core routers.

The figure below illustrates a typical situation in which an SP offers firewall services to VPN customers VPN1 and VPN2, thereby protecting VPN sites from the external network (for example, Shared Services and the Internet) and vice versa.

Figure 13 Distributed Network



In this example, VPN1 has two sites, Site A and Site B, that span across the MPLS core. Site A is connected to PE1, and Site B is connected to PE2. VPN2 has only one site that is connected to PE2.

Each VPN (VPN1 and VPN2) has the following:

- A VLAN segment in the Shared Service that is connected to the corresponding VLAN subinterface on PE3.
- Internet access through the PE3 router that is connected to the Internet

A distributed network requires the following firewall policies:

- VPN Firewall (VPN1-FW and VPN2-FW)--Inspects VPN-generated traffic that is destined to Shared Service or the Internet and blocks all non-firewall traffic that is coming from outside (Shared Service or the Internet), thereby protecting the VPN sites from outside traffic. This firewall typically is deployed on the VRF interface of the ingress PE router that is connected to the VPN site being protected. It is deployed in the inbound direction because the VRF interface is inbound to the VPN site being protected.
- Shared Service Firewall (SS-FW)--Inspects Shared Service-originated traffic that is destined to VPN sites and blocks all non-firewall traffic that is coming from outside (the VPN site), thereby protecting

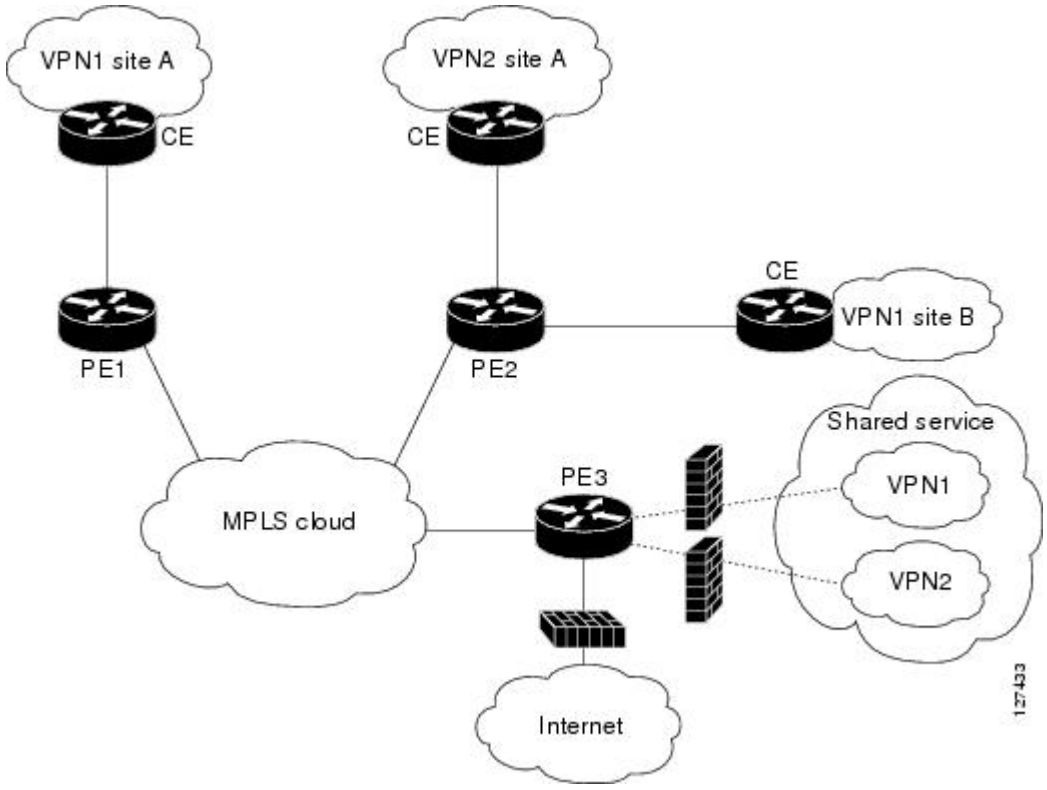
the Shared Service network from VPN sites. This firewall typically is deployed on the VRF interface of the ingress PE router that is connected to the VPN site from where the Shared Service is being protected. It is deployed in the outbound direction because the VRF interface is outbound to the Shared Service that is being protected.

- Generic-VPN Firewall (GEN-VPN-FW)--Inspects VPN-generated traffic that is destined to the Internet and blocks all non-firewall traffic that is coming from the Internet, thereby protecting all VPNs from the Internet. This firewall typically is deployed on the Internet-facing interface of the PE router that is connected to the Internet. It is deployed in the outbound direction because the Internet-facing interface is outbound to VPNs being protected.
- Internet Firewall (INET-FW)--Inspects Internet-generated traffic that is destined to Shared Service and blocks all non-firewall traffic that is coming from VPNs or Shared Service, thereby protecting the Internet from VPNs. This firewall typically is deployed on the Internet-facing interface of the PE router that is connected to the Internet. It is deployed in the inbound direction because the Internet-facing interface is inbound to the Internet being protected.

Hub-and-Spoke Network Inclusion of VRF Aware Cisco IOS Firewall

The figure below illustrates a hub-and-spoke network where the firewalls for all VPN sites are applied on the egress PE router PE3 that is connected to the Shared Service.

Figure 14 Hub-and-Spoke Network



Typically each VPN has a VLAN and/or VRF subinterface connected to the Shared Service. When a packet arrives from an MPLS interface, the inner tag represents the VPN-ID. MPLS routes the packet to the corresponding subinterface that is connected to Shared Service.

A Hub-and-Spoke network requires the following firewall policies:

- VPN Firewall (VPN1-FW and VPN2-FW)--Inspects VPN-generated traffic that is destined to Shared Service and blocks all non-firewall traffic that is coming from Shared Service, thereby protecting the VPN sites from Shared Service traffic. This firewall typically is deployed on the VLAN subinterface of the egress PE router that is connected to the Shared Service network. It is deployed in the outbound direction because the VLAN interface is outbound to the VPN site being protected.
- Shared Service Firewall (SS-FW)--Inspects Shared Service originated traffic that is destined to the VPN/Internet and blocks all non-firewall traffics that is coming from outside, thereby protecting the Shared Service network from VPN/Internet traffic. This firewall typically is deployed on the VLAN interface of the egress PE router that is connected to the Shared Service being protected. It is deployed in the inbound direction because the VLAN interface is inbound to the Shared Service being protected.
- Generic-VPN firewall (GEN-VPN-FW)--Inspects VPN-generated traffic that is destined to the Internet and blocks all non-firewall traffic that is coming from the Internet, thereby protecting all VPNs from the Internet. This firewall typically is deployed on the Internet-facing interface of the PE router that is connected to the Internet. It is deployed in the outbound direction because the Internet-facing interface is outbound to the VPNs being protected.
- Internet firewall (INET-FW)--Inspects Internet-generated traffic that is destined to Shared Service and blocks all non-firewall traffic that is coming from VPNs or Shared Service, thereby protecting the Internet from VPNs. This firewall typically is deployed on the Internet-facing interface of the PE router that is connected to the Internet. It is deployed in the inbound direction because the Internet-facing interface is inbound to the Internet being protected.

How to Configure VRF Aware Cisco IOS Firewall

Configuring and Checking ACLs to Ensure that Non-Firewall Traffic is Blocked

To configure ACLs and verify that only inspected traffic can pass through the firewall, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
4. **interface** *interface-type*
5. **ip access-group** {*access-list-number* | *access-list-name*} {**in** | **out**}
6. **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 access-list extended** *access-list-name* **Example:** `Router(config)# ip access-list extended vpn-ac1`	Defines an extended IP ACL to block non-firewall traffic in both inbound and outbound directions.
Step 4	**interface** *interface-type* **Example:** `Router(config)# interface ethernet0/1.10`	Enters interface configuration mode and specifies an interface that is associated with a VRF.
Step 5	**ip access-group** {*access-list-number* | *access-list-name*} {**in** | **out**} **Example:** `Router(config-if)# ip access-group vpn-acl in`	Controls access to an interface. Applies the previously defined IP access list to a VRF interface whose non-firewall traffic is blocked.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode. Returns to global configuration mode.

Creating and Naming Firewall Rules and Applying the Rules to the Interface

To create and name firewall rules and apply the rules to the interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect name** *inspection-name* [**parametermax-sessions***number*] *protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout***seconds*]
4. **interface** *interface-id*
5. **ip inspect** *rule-name* {**in** | **out**}
6. **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-name* [**parametermax-sessions***number*] *protocol* [**alert** {**on** | **off**}] [**audit-trail** {**on** | **off**}] [**timeout***seconds*] **Example:** `Router(config)# ip inspect name vpn_fw ftp`	Defines a set of inspection rules.
Step 4	**interface** *interface-id* **Example:** `Router(config)# interface ethernet0/1.10`	Enters interface configuration mode and specifies an interface that is associated with a VRF.
Step 5	**ip inspect** *rule-name* {**in** | **out**} **Example:** `Router(config-if)# ip inspect vpn_fw in`	Applies the previously defined inspection role to a VRF interface whose traffic needs to be inspected.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

Identifying and Setting Firewall Attributes

To identify and set firewall attributes, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip inspect tcp max-incomplete host** *number* **block-time** *minutes* [**vrf***vrf-name*]
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 tcp max-incomplete host** *number* **block-time** *minutes* [**vrf***vrf-name*] **Example:** `Router(config)# ip inspect tcp max-incomplete host 256 vrf bank-vrf`	Specifies threshold and blocking time values for TCP host-specific denial-of-service detection and prevention.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Verifying the VRF Aware Cisco IOS Firewall Configuration and Functioning

Verify the configuration and functioning of the firewall by entering the commands shown below.

SUMMARY STEPS

1. **show ip inspect** {**name***inspection-name* | **config** | **interfaces** | **session** [**detail**] | **statistics** | **all**}[**vrf***vrf-name*]
2. **show ip urlfilter** {**config** | **cache** | **statistics**} [**vrf***vrf-name*]

DETAILED STEPS

Step 1 **show ip inspect** {**name***inspection-name* | **config** | **interfaces** | **session** [**detail**] | **statistics** | **all**}[**vrf***vrf-name*]
Use this command to view the firewall configurations, sessions, statistics, and so forth, pertaining to a specified VRF. For example, to view the firewall sessions pertaining to the VRF bank, enter the following command:

Example:

```
Router# show ip inspect interfaces vrf bank
```

Step 2 **show ip urlfilter** {**config** | **cache** | **statistics**} [**vrf***vrf-name*]
Use this command to view the configurations, cache entries, statistics, and so forth, pertaining to a specified VRF. For example, to view the URL filtering statistics pertaining to the VRF bank, enter the following command:

Example:

```
Router# show ip urlfilter statistics vrf bank
```

Configuration Examples for VRF Aware Cisco IOS Firewall

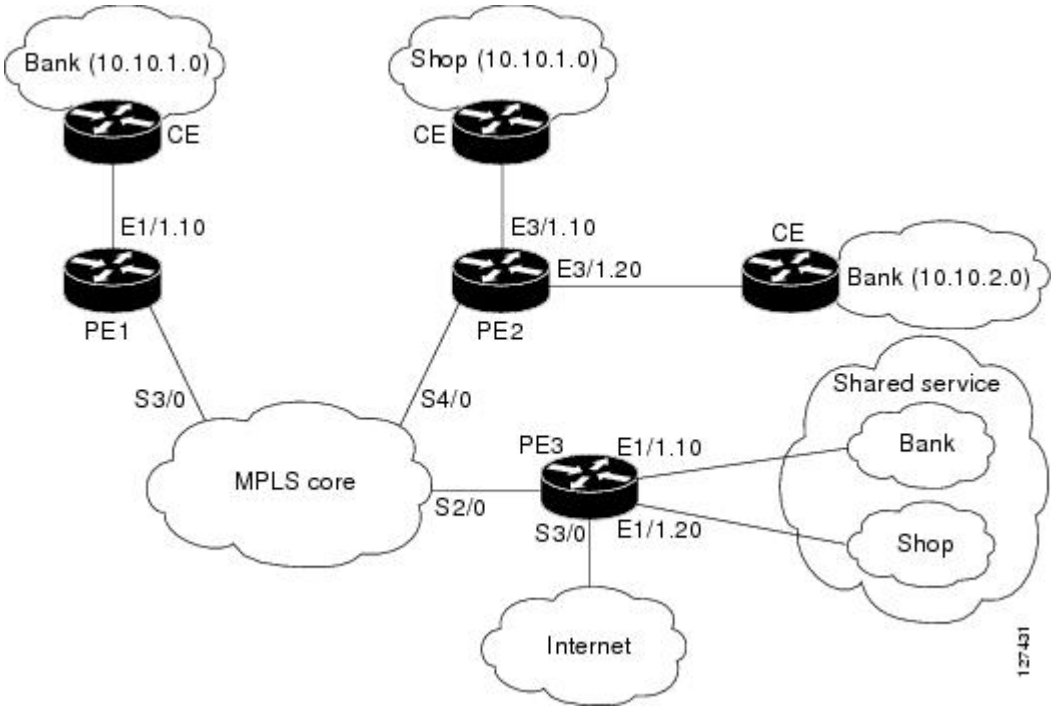
In the example illustrated in the figure below, a service provider offers firewall service to VPN customers Bank and Shop. The Bank VPN has the following two sites in an MPLS network:

- Site connected to PE1, whose network address is 10.10.1.0/24
- Site connected to PE2, whose network address is 10.10.2.0/24

The Bank VPN also has a VLAN network segment in Shared Service that is connected to PE3.

The Shop VPN has only one site, which is connected to PE4. The network address 10.10.1.0/24 is the same network address to which the Bank VPN site is connected.

Figure 15 VPN with Two Sites Across MPLS Network



Each VPN needs the following two firewalls:

- VPN firewall to protect the VPN site from Shared Services
- Shared Service (SS) firewall to protect SS from the VPN site

In addition, the following two firewalls are required:

- Internet firewall to protect VPNs from the Internet
- Generic VPN firewall to protect the Internet from VPNs

In this example, the security policies for Bank and Shop VPNs are as follows:

- Bank VPN Firewall--bank_vpn_fw (Inspects FTP, HTTP, and ESMTP protocols)
- Bank SS Firewall--bank_ss_fw (Inspects ESMTP protocol)
- Shop VPN Firewall--shop_vpn_fw (Inspects HTTP and RTSP protocols)
- Shop SS Firewall--shop_ss_fw (Inspects H323 protocol)

The security policies for the Internet firewall and generic VPN firewall are as follows:

- Internet firewall--inet_fw (Inspects HTTP and ESMTP protocols)
- Generic VPN firewall--gen_vpn_fw (Inspects FTP, HTTP, ESMTP, and RTSP protocols)

DISTRIBUTED NETWORK

PE1:

```
! VRF instance for the Bank VPN
ip vrf bank
rd 100:10
route-target export 100:10
route-target import 100:10

!
! VPN Firewall for Bank VPN protects Bank VPN from Shared Service
ip inspect name bank_vpn_fw ftp
ip inspect name bank_vpn_fw http
ip inspect name bank_vpn_fw esmtp

!
! Shared Service firewall for Bank VPN protects Shared Service from Bank VPN
ip inspect name bank_ss_fw esmtp

!
! VRF interface for the Bank VPN
interface ethernet0/1.10

!
! description of VPN site Bank to PE1
encapsulation dot1Q 10
ip vrf forwarding bank
ip address 10.10.1.2 255.255.255.0
ip access-group bank_ss_acl in
ip access-group bank_vpn_acl out
ip inspect bank_vpn_fw in
ip inspect bank_ss_fw out

!
! MPLS interface
interface Serial3/0
 ip unnumbered Loopback0

 tag-switching ip

 serial restart-delay 0

!
! ACL that protects the VPN site Bank from Shared Service
ip access-list extended bank_vpn_acl

 permit ip 10.10.2.0 0.0.0.255 10.10.1.0 0.0.0.255

 permit tcp any any eq smtp

 deny   ip  any any log
!
! ACL that protects Shared Service from VPN site Bank
ip access-list extended bank_ss_acl

 permit ip 10.10.1.0 0.0.0.255 10.10.2.0 0.0.0.255

 permit tcp any any eq ftp

 permit tcp any any eq http
 permit tcp any any eq smtp

 deny   ip  any any log
```

PE2:

```
! VRF instance for the Bank VPN
ip vrf bank
```

```
rd 100:10
route-target export 100:10
route-target import 100:10

!
! VRF instance for the Shop VPN
ip vrf shop
rd 200:20
route-target export 200:20
route-target import 200:20

!
! VPN firewall for Bank BPN protects Bank VPN from Shared Service
ip inspect name bank_vpn_fw ftp
ip inspect name bank_vpn_fw http
ip inspect name bank_vpn_fw esmtp

!
! Shared Service firewall for Bank VPN protects Shared Service from Bank VPN
ip inspect name bank_ss_fw esmtp

!
! VPN firewall for Shop VPN protects Shop VPN from Shared Service
ip inspect name shop_vpn_fw http
ip inspect name shop_vpn_fw rtsp

!
! Shared Service firewall for Shop VPN protects Shared Service from Shop VPN
ip inspect name shop_ss_fw h323
!
! VRF interface for the Bank VPN
interface Ethernet3/1.10

!
! description of VPN site Bank to PE2
encapsulation dot1Q 10
ip vrf forwarding bank
ip address 10.10.2.2 255.255.255.0
ip access-group bank_ss_acl in
ip access-group bank_vpn_acl out
ip inspect bank_vpn_fw in
ip inspect bank_ss_fw out

!
interface Ethernet3/1.20

!
! description of VPN site Shop to PE2
encapsulation dot1Q 20
ip vrf forwarding shop
ip address 10.10.1.2 255.255.255.0
ip access-group shop_ss_acl in
ip access-group shop_vpn_acl out
ip inspect shop_vpn_fw in
ip inspect shop_ss_fw out
interface Serial4/0

 ip unnumbered Loopback0

 tag-switching ip

 serial restart-delay 0

!
! ACL that protects the VPN site Bank from Shared Service
ip access-list extended bank_vpn_acl

 permit ip 10.10.1.0 0.0.0.255 10.10.2.0 0.0.0.255

 permit tcp any any eq smtp

 deny   ip  any any log
```

```
!
! ACL that protects Shared Service from VPN site Bank
ip access-list extended bank_ss_acl
 permit ip 10.10.2.0 0.0.0.255 10.10.1.0 0.0.0.255

 permit tcp any any eq ftp

 permit tcp any any eq http

 permit tcp any any eq smtp

 deny   ip  any any log

!
! ACL that protects VPN site Shop from Shared Service
ip access-list extended shop_vpn_acl

 permit tcp any any eq h323

 deny   ip any any log

!
ip access-list extended shop_ss_acl

 permit tcp any any eq http

 permit tcp any any eq rtsp
deny   ip  any any log
```

PE3:

```
! VRF instance for the Bank VPN
ip vrf bank
rd 100:10
route-target export 100:10
route-target import 100:10

!
! VRF instance for the Shop VPN
ip vrf shop
rd 200:20
route-target export 200:20
route-target import 200:20

!
! Generic VPN firewall to protect Shop and Bank VPNs from internet
ip inspect name gen_vpn_fw esmtp
ip inspect name gen_vpn_fw ftp
ip inspect name gen_vpn_fw http
ip inspect name gen_vpn_fw rtsp

!
! Internet firewall to prevent malicious traffic from being passed
! to internet from Bank and Shop VPNs
ip inspect name inet_fw esmtp
ip inspect name inet_fw http

!
! VRF interface for the Bank VPN
interface Ethernet1/1.10

!
! Description of Shared Service to PE3
encapsulation dot1Q 10
ip vrf forwarding bank
ip address 10.10.1.50 255.255.255.0

!
! VRF interface for the Shop VPN
interface Ethernet1/1.20
```

```
!
! Description of Shared Service to PE3
encapsulation dot1Q 20
ip vrf forwarding shop
ip address 10.10.1.50 255.255.255.0
interface Serial2/0

 ip unnumbered Loopback0

 tag-switching ip

 serial restart-delay 0

!
! VRF interface for the Bank VPN
interface Serial3/0

!
! Description of Internet-facing interface
ip address 192.168.10.2 255.255.255.0
ip access-group inet_acl out
ip access-group gen_vpn_acl in
ip inspect gen_vpn_fw out
ip inspect inet_fw in

!
! ACL that protects the Bank and Shop VPNs from internet
ip access-list extended gen_vpn_acl

 permit tcp any any eq smtp

 permit tcp any any eq www

 deny   ip  any any log

!
! ACL that protects internet from Bank and Shop VPNs
ip access-list extended inet_acl

 permit tcp any any eq ftp

 permit tcp any any eq http

 permit tcp any any eq smtp

 permit tcp any any eq rtsp

 deny   ip  any any log
```

HUB-AND-SPOKE NETWORK

PE3:

```
! VRF instance for the VPN Bank
ip vrf bank
rd 100:10
route-target export 100:10
route-target import 100:10

!
! VRF instance for the VPN Shop
ip vrf shop
rd 200:20
route-target export 200:20
route-target import 200:20

!
! VPN firewall for Bank BPN protects Bank VPN from Shared Service
ip inspect name bank_vpn_fw ftp
ip inspect name bank_vpn_fw http
```

```
ip inspect name bank_vpn_fw esmtp

!
! Shared Service firewall for Bank VPN protects Shared Service from Bank VPN
ip inspect name bank_ss_fw esmtp

!
! VPN firewall for Shop VPN protects Shop VPN from Shared Service
ip inspect name shop_vpn_fw http
ip inspect name shop_vpn_fw rtsp

!
! Shared Service firewall for Shop VPN protects Shared Service from Shop VPN
ip inspect name shop_ss_fw h323

!
! Generic VPN firewall protects Shop and Bank VPNs from internet
ip inspect name gen_vpn_fw esmtp
ip inspect name gen_vpn_fw ftp
ip inspect name gen_vpn_fw http
ip inspect name gen_vpn_fw rtsp

!
! Internet firewall prevents malicious traffic from being passed
! to internet from Bank and Shop VPNs
ip inspect name inet_fw esmtp
ip inspect name inet_fw http

!
! VRF interface for the Bank VPN
interface Ethernet1/1.10

!
! description of Shared Service to PE3
encapsulation dot1Q 10
ip vrf forwarding bank
ip address 10.10.1.50 255.255.255.0
ip access-group bank_ss_acl out
ip access-group bank_vpn_acl in
ip inspect bank_vpn_fw out
ip inspect bank_ss_fw in

!
! VRF interface for the Shop VPN
interface Ethernet1/1.20
!
! description of Shared Service to PE3
encapsulation dot1Q 20
ip vrf forwarding shop
ip address 10.10.1.50 255.255.255.0
ip access-group shop_ss_acl out
ip access-group shop_vpn_acl in
ip inspect shop_vpn_fw out
ip inspect shop_ss_fw in
interface Serial2/0

 ip unnumbered Loopback0

 tag-switching ip

 serial restart-delay 0
!
! VRF interface for the Bank VPN
interface Serial3/0

!
! description of Internet-facing interface
ip address 192.168.10.2 255.255.255.0
ip access-group inet_acl out
ip access-group gen_vpn_acl in
ip inspect gen_vpn_fw out
ip inspect inet_fw in
```

```
!
! ACL that protects the VPN site Bank from Shared Service
ip access-list extended bank_vpn_acl

 permit tcp any any eq smtp

 deny   ip  any any log
!
! ACL that protects Shared Service from VPN site Bank
ip access-list extended bank_ss_acl

 permit tcp any any eq ftp

 permit tcp any any eq http

 permit tcp any any eq smtp

 deny   ip  any any log

!
! ACL that protects VPN site Shop from Shared Service
ip access-list extended shop_vpn_acl

 permit tcp any any eq h323

 deny   ip any any log

!
ip access-list extended shop_ss_acl

 permit tcp any any eq http
 permit tcp any any eq rtsp
 deny   ip  any any log
!
! ACL that protects the Bank and Shop VPNs from internet
ip access-list extended gen_vpn_acl

 permit tcp any any eq smtp

 permit tcp any any eq www
 deny   ip  any any log
!
! ACL that protects internet from Bank and Shop VPNs
ip access-list extended inet_acl

 permit tcp any any eq ftp
 permit tcp any any eq http

 permit tcp any any eq smtp

 permit tcp any any eq rtsp

 deny   ip  any any log
```

In the example illustrated in the figure below, the Cisco IOS Firewall is configured on PE1 on the VRF interface E3/1. The host on NET1 wants to reach the server on NET2.

Figure 16 Sample VRF Aware Cisco IOS Firewall Network

The configuration steps are followed by a sample configuration and log messages.

1. Configure VRF on PE routers.
2. Ensure that your network supports MPLS traffic engineering.
3. Confirm that the VRF interface can reach NET1 and NET2.
4. Configure the VRF Aware Cisco IOS Firewall.
   a. Configure and apply ACLs.
   b. Create Firewall rules and apply them to the VRF interface.
5. Check for VRF firewall sessions.

VRF Configuration on PE1

```
! configure VRF for host1
ip cef
ip vrf vrf1
rd 100:1
route-target export 100:1
route-target import 100:1
exit
end
!
! apply VRF to the interface facing CE
interface ethernet3/1
ip vrf forwarding vrf1
ip address 190.1.1.2 255.255.0.0
!
! make the interface facing the MPLS network an MPLS interface
interface serial2/0
mpls ip
ip address 191.171.151.1 255.255.0.0
!
! configure BGP protocol for MPLS network
router bgp 100
no synchronization
bgp log-neighbor-changes
neighbor 191.171.151.2 remote-as 100
neighbor 191.171.151.2 update-source serial2/0
no auto-summary
address-family vpnv4
neighbor 191.171.151.2 activate
neighbor 191.171.151.2 send-community both
exit-address-family
address-family ipv4 vrf vrf1
redistribute connected
redistribute static
no auto-summary
no synchronization
exit-address-family
!
! configure VRF static route to reach CE network
ip route vrf vrf1 192.168.4.0 255.255.255.0 190.1.1.1
```

VRF Configuration on PE2

```
! configure VRF for host2
ip cef
ip vrf vrf1
rd 100:1
route-target export 100:1
route-target import 100:1
!
! apply VRF on CE-facing interface
interface fastethernet0/0
ip vrf forwarding vrf1
ip address 193.1.1.2 255.255.255.0
!
! make MPLS network-facing interface an MPLS interface
interface serial1/0
mpls ip
ip address 191.171.151.2 255.255.0.0
!
! configure BGP protocol for MPLS network
router bgp 100
no synchronization
bgp log-neighbor-changes
neighbor 191.171.151.1 remote-as 100
neighbor 191.171.151.1 update-source serial1/0
no auto-summary
address-family vpnv4
neighbor 191.171.151.1 activate
neighbor 191.171.151.1 send-community both
```

```
exit-address-family
address-family ipv4 vrf vrf1
redistribute connected
redistribute static
no auto-summary
no synchronization
exit-address-family
!configure VRF static route to reach CE network
ip route vrf vrf1 192.168.4.0 255.255.255.0 193.1.1.1
```

Configuration on CE1

```
interface e0/1
ip address 190.1.1.1 255.255.255.0
interface e0/0
ip address 192.168.4.2 255.255.255.0
ip route 192.168.104.0 255.255.255.0 190.1.1.2
```

Configuration on CE2

```
interface e0/1
ip address 190.1.1.1 255.255.255.0
interface e0/0
ip address 192.168.4.2 255.255.255.0
ip route 192.168.4.0 255.255.255.0 193.1.1.2
```

Configure Firewall on PE1 and Apply on the VRF Interface

```
! configure ACL so that NET2 cannot access NET1
ip access-list extended 105
permit tcp any any fragment
permit udp any any fragment
deny tcp any any
deny udp any any
permit ip any any
!
! apply ACL to VRF interface on PE1
interface ethernet3/1
ip access-group 105 out
!
! configure firewall rule
ip inspect name test tcp
!
! apply firewall rule on VRF interface
interface ethernet3/1
 ip inspect test in
```

Check for VRF Firewall Sessions When Host on NET1 Tries to Telnet to Server on NET2

```
show ip inspect session vrf vrf1
Established Sessions
 Session 659CE534 (192.168.4.1:38772)=>(192.168.104.1:23) tcp SIS_OPEN
!
! checking for ACLs
show ip inspect session detail vrf vrf1 | include ACL 105
  Out SID 192.168.104.1[23:23]=>192.168.4.1[38772:38772] on ACL 105
(34 matches)
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
VRF-lite	*Catalyst 4500 Series Switch Cisco IOS Software Configuration Guide , Release 12.2*
MPLS VPN	*Configuring a Basic MPLS VPN , Document ID 13733*
VRF Aware IPSec	• *VRF-Aware IPSec* feature module, Release 12.2(15)T • *Cisco IOS Security Configuration Guide* , Release 12.3 • *Cisco IOS Security Command Reference* , Release 12.3T
VRF management	*Cisco 12000/10720 Router Manager User's Guide , Release 3.2*
NAT	• *NAT and Stateful Inspection of Cisco IOS Firewall* , White Paper • *Configuring Network Address Translation: Getting Started* --Document ID 13772

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 VRF Aware Cisco 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 16 Feature Information for VRF Aware Cisco IOS Firewall

Feature Name	Releases	Feature Information
VRF Aware Cisco IOS Firewall	12.3(14)T	VRF Aware Cisco IOS Firewall applies Cisco IOS Firewall functionality to VRF (Virtual Routing and Forwarding) interfaces when the firewall is configured on a service provider (SP) or large enterprise edge router. SPs can provide managed services to small and medium business markets. The VRF Aware Cisco IOS Firewall supports VRF-aware URL filtering and VRF-lite (also known as Multi-VRF CE). The following commands were introduced or modified:**clearipurlfiltercache**, **ipinspectalert-off**, **ipinspectaudittrail**, **ipinspectdns-timeout**, **ipinspectmax-incompletehigh**, **ipinspectmax-incompletelow**, **ipinspectname**, **ipinspectone-minutehigh**, **ipinspectone-minutelow**, **ipinspecttcpfinwait-time**, **ipinspecttcpidle-time**, **ipinspecttcpmax-incompletehost**, **ipinspecttcpsynwait-time**, **ipinspectudpidle-time**, **ipurlfilteralert**, **ipurlfilterallowmode**, **ipurlfilteraudit-trail**, **ipurlfiltercache**, **ipurlfilterexclusive-domain**, **ipurlfilterexclusive-domain**, **ipurlfiltermax-request**, **ipurlfiltermax-resp-pak**, **ipurlfilterservervendor**, **ipurlfilterurlf-server-log**, **showipinspect**, **showipurlfiltercache**, **showipurlfilterconfig**, **showipurlfilterstatistics**.

Glossary

CE router --customer edge router. A router that is part of a customer network and that interfaces to a provider edge (PE) router.

CBAC --Context-Based Access Control. A protocol that provides internal users with secure access control for each application and for all traffic across network perimeters. CBAC enhances security by scrutinizing both source and destination addresses and by tracking each application's connection status.

data authentication --Refers to one or both of the following: data integrity, which verifies that data has not been altered, or data origin authentication, which verifies that the data was actually sent by the claimed sender.

data confidentiality --A security service where the protected data cannot be observed.

edge router --A router that turns unlabeled packets into labeled packets, and vice versa.

firewall --A router or access server, or several routers or access servers, designated as a buffer between any connected public networks and a private network. A firewall router uses access lists and other methods to ensure the security of the private network.

inspection rule --A rule that specifies what IP traffic (which application-layer protocols) will be inspected by CBAC at an interface.

intrusion detection --The Cisco IOS Firewall's Intrusion Detection System (Cisco IOS IDS) identifies the most common attacks, using signatures to detect patterns of misuse in network traffic.

IPSec --IP Security Protocol. A framework of open standards developed by the Internet Engineering Task Force (IETF). IPSec provides security for transmission of sensitive data over unprotected networks such as the Internet.

managed security services --A comprehensive set of programs that enhance service providers’ abilities to meet the growing demands of their enterprise customers. Services based on Cisco solutions include managed firewall, managed VPN (network based and premises based), and managed intrusion detection.

NAT --Network Address Translation. Translates a private IP address used inside the corporation to a public, routable address for use outside of the corporation, such as the Internet. NAT is considered a one-to-one mapping of addresses from private to public.

PE router --provider edge router. A router that is part of a service provider's network and is connected to a customer edge (CE) router.

skinny --Skinny Client Control Protocol (SCCP). A protocol that enables CBAC to inspect Skinny control packets that are exchanged between a Skinny client and the Call Manager (CM); CBAC then configures the router (also known as the Cisco IOS Firewall) to enable the Skinny data channels to traverse through the router.

traffic filtering --A capability that allows you to 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.

traffic inspection --CBAC inspection of 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 (sessions that originated from within the protected internal network).

UDP -- User Datagram Protocol. Connectionless transport layer protocol in the TCP/IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery, requiring that error processing and retransmission be handled by other protocols.

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

vrf --A VPN routing/forwarding instance. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a provider edge (PE) router.

VRF table --A table that stores routing data for each VPN. The VRF table defines the VPN membership of a customer site attached to the network access server (NAS). Each VRF table comprises an IP routing table, a derived Cisco Express Forwarding (CEF) table, and guidelines and routing protocol parameters that control the information that is included in the routing table.

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