

# 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."

For a complete description of the CBAC commands used in this chapter, refer to the "Context-Based Access Control Commands" chapter in the *Security Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

## CBAC Overview

This section describes:

- What CBAC Does
- What CBAC Does Not Do
- How CBAC Works
- When and Where to Configure CBAC
- The CBAC Process
- Supported Protocols
- Restrictions
- Memory and Performance Impact

## What CBAC Does

CBAC intelligently filters TCP and UDP packets based on application-layer protocol session information and can be used for intranets, extranets and the Internet. 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. (In other words, CBAC can inspect traffic for sessions that originate from the external network.) However, while this example discusses inspecting traffic for sessions that originate from the external network, CBAC can inspect traffic for sessions that originate from either side of the firewall.

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 TCP or UDP 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.

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 (sessions that originated from within the protected internal network).

CBAC also provides the following benefits:

- Java blocking
- Denial-of-Service prevention and detection
- Real-time alerts and audit trails

## 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. CBAC only detects and protects against attacks that travel through the firewall.

CBAC protects against certain attacks but 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

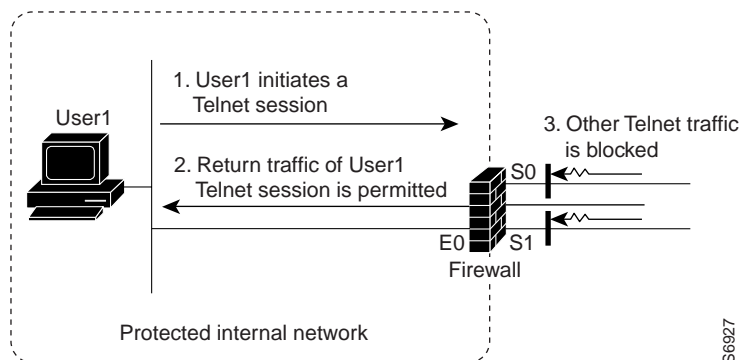
You should understand the material in this section before you configure CBAC. If you do not understand how CBAC works, you might inadvertently introduce security risks by configuring CBAC inappropriately.

### 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.

In Figure 9, 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 9 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. For these protocols, packets flowing through the firewall in any direction are inspected, as long as they flow through the interface where inspection is configured.

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

CBAC inspects and monitors only the control channels of connections; the data channels are not inspected. For example, during FTP sessions both the control and data channels (which are created when a data file is transferred) are monitored for state changes, but only the control channel is inspected (that is, the CBAC software parses the FTP commands and responses).

CBAC inspection recognizes application-specific commands in the control channel, and detects and prevents certain application-level attacks.

### 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 packet's connection.

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. Inspection controls the traffic that belongs to a valid session and forwards the traffic it does not know. 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, similar 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 Firewall feature set configured as described previously in the section “The Cisco IOS Firewall Feature Set.”

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

- 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.

In rare cases, you might want to configure CBAC in two directions at one or more interface, which is a more complex solution. CBAC is usually only configured in two directions when the networks on both sides of the firewall should be protected, such as with extranet or intranet configurations. 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 other 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 Telnet 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 “Define an Inspection Rule” for configuring CBAC inspection information.)

- 4 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.
- 5 The outbound packet is forwarded out the interface.
- 6 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.
- 7 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.
- 8 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.
- 9 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.

## 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)
- Java
- UNIX R-commands (such as rlogin, rexec, and rsh)
- RealAudio

- RPC (Sun RPC, not DCE RPC or Microsoft RPC)
- SMTP
- SQL\*Net
- StreamWorks
- TFTP
- VDOLive

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

## Restrictions

CBAC is available only for IP protocol traffic. Only TCP and UDP packets are inspected. (Other IP traffic, such as ICMP, cannot be filtered with CBAC and should be filtered with basic access lists instead.)

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.

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 or evaluated by access lists.

CBAC ignores ICMP Unreachable messages.

## 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.

## Cisco Encryption Technology and CBAC Compatibility

If encrypted traffic is exchanged between two routers, and the firewall is in between the two routers, CBAC might not work as anticipated. This is because the packets' payloads are encrypted, so CBAC cannot accurately inspect the payloads.

Also, if both encryption and CBAC are configured at the same firewall, CBAC will not work for certain protocols. In this case, CBAC will work with single-channel TCP and UDP, except for Java and SMTP. But CBAC will not work with multichannel protocols, except for StreamWorks and CU-SeeMe. So if you configure encryption at the firewall, you should configure CBAC for only the following protocols:

- Generic TCP
- Generic UDP

- CU-SeeMe
- StreamWorks

### IPSec and CBAC Compatibility

When CBAC and IPSec are enabled on the same router, and the target 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.

## Memory and Performance Impact

Using 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).

## CBAC Configuration Task List

To configure CBAC, perform the tasks described in the following sections:

- Pick an Interface: Internal or External
- Configure IP Access Lists at the Interface
- Configure Global Timeouts and Thresholds
- Define an Inspection Rule
- Apply the Inspection Rule to an Interface

You can also perform the tasks described in the following sections. These tasks are optional.

- Display Configuration, Status, and Statistics for Context-Based Access Control
- Debug Context-Based Access Control
- Interpret Syslog and Console Messages Generated by Context-Based Access Control
- Turn Off Context-Based Access Control

---

**Note** 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.

---

For CBAC configuration examples, see the “CBAC Configuration Example” section located at the end of this chapter.

## Pick an Interface: Internal or External

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

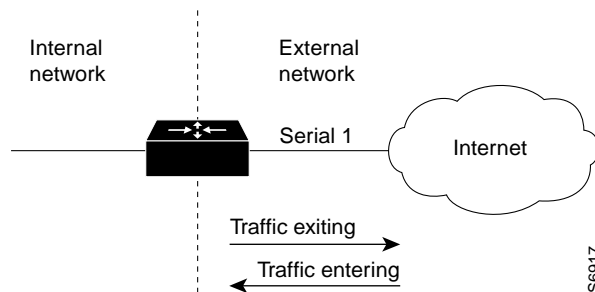
“Internal” refers to the side where sessions must originate for their traffic to be permitted through the firewall. “External” refers to the side where sessions cannot originate (sessions originating from the external side will be blocked).

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 is rarely configured in two directions, and usually only when the firewall is between two networks that need protection from each other, such as with two partners’ networks connected by the firewall.)

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.

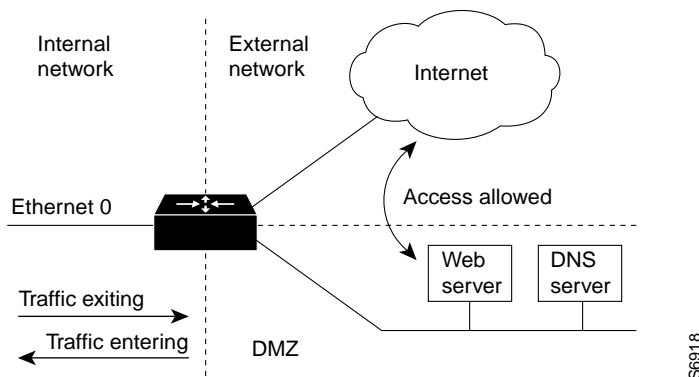
The first topology is shown in Figure 10. In this simple topology, CBAC is configured for the *external* interface Serial 1. This prevents specified protocol traffic from entering the firewall and the internal network, unless the traffic is part of a session initiated from within the internal network.

**Figure 10 Simple Topology—CBAC Configured at the External Interface**



The second topology is shown in Figure 11. 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 11 DMZ Topology—CBAC Configured at the Internal Interface**



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

## Configure 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 two general rules when evaluating your IP access lists at the firewall:

- Permit CBAC traffic leaving 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.

---

### External Interface

Here are some tips 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 *Network Protocols Configuration Guide, Part 1*.

### 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 Cisco IOS Release 11.3 *Network Protocols Configuration Guide, Part 1*.

## Configure 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. Note that 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 **ip inspect tcp max-incomplete host** command (see the last row in the following table).

All the available CBAC timeouts and thresholds are listed in the following table, along with the corresponding command and default value.

To change a global timeout or threshold listed in the “Timeout or Threshold Value to Change” column, use the global configuration command in the “Command” column:

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.	<b>ip inspect tcp synwait-time</b> <i>seconds</i>	30 seconds
The length of time a TCP session will still be managed after the firewall detects a FIN-exchange.	<b>ip inspect tcp finwait-time</b> <i>seconds</i>	5 seconds
The length of time a TCP session will still be managed after no activity (the TCP idle timeout). <sup>1</sup>	<b>ip inspect tcp idle-time</b> <i>seconds</i>	3600 seconds (1 hour)
The length of time a UDP session will still be managed after no activity (the UDP idle timeout). <sup>1</sup>	<b>ip inspect udp idle-time</b> <i>seconds</i>	30 seconds
The length of time a DNS name lookup session will still be managed after no activity.	<b>ip inspect dns-timeout</b> <i>seconds</i>	5 seconds
The number of existing half-open sessions that will cause the software to start deleting half-open sessions. <sup>2</sup>	<b>ip inspect max-incomplete high</b> <i>number</i>	500 existing half-open sessions
The number of existing half-open sessions that will cause the software to stop deleting half-open sessions. <sup>2</sup>	<b>ip inspect max-incomplete low</b> <i>number</i>	400 existing half-open sessions
The rate of new unestablished sessions that will cause the software to start deleting half-open sessions. <sup>2</sup>	<b>ip inspect one-minute high</b> <i>number</i>	500 half-open sessions per minute

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

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

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

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

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

## 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. Measurements are made once a minute.

When the number of existing half-open sessions rises above a threshold (the **max-incomplete high** 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-incomplete low** number).

When the rate of new connection attempts rises above a threshold (the **one-minute high** 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-minute low** number). The rate thresholds are measured as the number of new session connection attempts detected in the last one-minute sample period. (The rate is calculated as an exponentially-decayed rate.)

## Define 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.

To define an inspection rule, follow the instructions in the following sections:

- Configure Application-Layer Protocol Inspection
- Configure Generic TCP and UDP Inspection

## Configure Application-Layer Protocol Inspection

---

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

---

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

Command	Purpose
<b>ip inspect name</b> <i>inspection-name protocol</i> [ <b>timeout</b> <i>seconds</i> ]	Configure CBAC inspection for an application-layer protocol (except for RPC and Java). Use one of the protocol keywords defined in Table 18.  Repeat this command for each desired protocol. Use the same <i>inspection-name</i> to create a single inspection rule.
<b>ip inspect name</b> <i>inspection-name rpc program-number number</i> [ <b>wait-time</b> <i>minutes</i> ] [ <b>timeout</b> <i>seconds</i> ]	Enable 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 <i>inspection-name</i> to create a single inspection rule.

Refer to the description of the **ip inspect name (global configuration)** command in the “Context-Based Access Control Commands” chapter in the *Security Command Reference* for complete information about how the command works with each application-layer protocol.

To enable CBAC inspection for Java, see the following section, “Configure Java Inspection.”

Table 18 identifies application protocol keywords.

**Table 18 Application Protocol Keywords**

Application Protocol	<i>protocol</i> Keyword
CU-SeeMe	<b>cuseeme</b>
FTP	<b>ftp</b>
H.323	<b>h323</b>
UNIX R commands (rlogin, rexec, rsh)	<b>remd</b>
RealAudio	<b>realaudio</b>
SMTP	<b>smtp</b>

**Table 18** Application Protocol Keywords (Continued)

Application Protocol	<i>protocol</i> Keyword
SQL*Net	<b>sqlnet</b>
StreamWorks	<b>streamworks</b>
TFTP	<b>tftp</b>
VDOLive	<b>vdolive</b>

### Configure Java Inspection

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 agreeable solution, you can use CBAC 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.

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.)

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

Step	Command	Purpose
1	<b>ip access-list standard</b> <i>name</i> <b>permit</b> ... <b>deny</b> ... (Use <b>permit</b> and <b>deny</b> statements as appropriate.) or <b>access-list</b> <i>access-list-number</i> { <b>deny</b>   <b>permit</b> } <i>source</i> [ <i>source-wildcard</i> ]	Create a standard access list that permits traffic only from friendly sites, and denies traffic from hostile sites.  If you want all internal users to be able to download friendly applets, use the <b>any</b> keyword for the destination as appropriate—but be careful to not misuse the <b>any</b> keyword to inadvertently allow all applets through.
2	<b>ip inspect name</b> <i>inspection-name</i> <b>http</b> [ <b>java-list</b> <i>access-list</i> ] [ <b>timeout</b> <i>seconds</i> ]	Block 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.  Use the same <i>inspection-name</i> as when you specified other protocols, to create a single inspection rule.



**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.

## Configure 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 **ip inspect udp idle-time** command.)

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

Command	Purpose
<b>ip inspect name</b> <i>inspection-name</i> <b>tcp</b> [ <b>timeout</b> <i>seconds</i> ]	Enable CBAC inspection for TCP packets. Use the same <i>inspection-name</i> as when you specified other protocols, to create a single inspection rule.
<b>ip inspect name</b> <i>inspection-name</i> <b>udp</b> [ <b>timeout</b> <i>seconds</i> ]	Enable CBAC inspection for UDP packets. Use the same <i>inspection-name</i> as when you specified other protocols, to create a single inspection rule.

## Apply 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 interface configuration command:

Command	Purpose
<b>ip inspect</b> <i>inspection-name</i> { <b>in</b>   <b>out</b> }	Apply an inspection rule to an interface.

## Display Configuration, Status, and Statistics for Context-Based Access Control

You can view certain CBAC information by using one or more of the following EXEC commands:

Command	Purpose
<b>show ip inspect name</b> <i>inspection-name</i>	Show a particular configured inspection rule.
<b>show ip inspect config</b>	Show the complete CBAC inspection configuration.
<b>show ip inspect interfaces</b>	Show interface configuration with regards to applied inspection rules and access lists.
<b>show ip inspect session</b> [detail]	Show existing sessions that are currently being tracked and inspected by CBAC.
<b>show ip inspect all</b>	Show all CBAC configuration and all existing sessions that are currently being tracked and inspected by CBAC.

## Debug 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.

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

Command	Purpose
<b>ip inspect audit trail</b>	Turn on CBAC audit trail messages.

If required, you can also use the CBAC **debug** commands listed in this section. (Debugging can be turned off for each of the commands in this section by using the **no** form of the command. To disable all debugging, use the privileged EXEC commands **no debug all** or **undebug all**.)

The available **debug** commands are listed in the following categories:

- Generic Debug Commands
- Transport Level Debug Commands
- Application Protocol Debug Commands

For a complete description of the debug commands, refer to the *Debug Command Reference*.

### Generic Debug Commands

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

Command	Purpose
<b>debug ip inspect function-trace</b>	Display messages about software functions called by CBAC.
<b>debug ip inspect object-creation</b>	Display messages about software objects being created by CBAC. Object creation corresponds to the beginning of CBAC-inspected sessions.
<b>debug ip inspect object-deletion</b>	Display messages about software objects being deleted by CBAC. Object deletion corresponds to the closing of CBAC-inspected sessions.
<b>debug ip inspect events</b>	Display messages about CBAC software events, including information about CBAC packet processing.

Command	Purpose
<b>debug ip inspect timers</b>	Display messages about CBAC timer events such as when a CBAC idle timeout is reached.
<b>debug ip inspect detail</b>	Enable 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
<b>debug ip inspect tcp</b>	Display messages about CBAC-inspected TCP events, including details about TCP packets.
<b>debug ip inspect udp</b>	Display 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
<b>debug ip inspect <i>protocol</i></b>	Display messages about CBAC-inspected protocol events, including details about the protocol's packets. Refer to Table 19 to determine the protocol keyword.

Table 19 identifies application protocol keywords for the **debug ip inspect** command

**Table 19 Application Protocol Keywords for the debug ip inspect Command**

Application Protocol	<i>protocol</i> keyword
CU-SeeMe	<b>cuseeme</b>
FTP commands and responses	<b>ftp-cmd</b>
FTP tokens (enables tracing of the FTP tokens parsed)	<b>ftp-tokens</b>
H.323	<b>h323</b>
Java applets	<b>http</b>
UNIX R commands (rlogin, rexec, rsh)	<b>rcmd</b>
RealAudio	<b>realaudio</b>
RPC	<b>rpc</b>
SMTP	<b>smtp</b>
SQL*Net	<b>sqlnet</b>
StreamWorks	<b>streamworks</b>
TFTP	<b>tftp</b>
VDOLive	<b>vdolive</b>

## Interpret Syslog and Console Messages Generated by Context-Based Access Control

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.

The following types of error messages can be generated by CBAC:

- Denial-of-Service Attack Detection Error Messages
- SMTP Attack Detection Error Message
- Java Blocking Error Message
- FTP Error Messages
- Audit Trail Error Message

For explanations and recommended actions related to the error messages mentioned in this chapter, refer to the *Software 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 above 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 Message

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)
```

### Java Blocking Error Message

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 Error Message

CBAC provides audit trail messages to record details about inspected sessions. 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
```

## Turn Off Context-Based Access Control

You can turn off CBAC, with the **no ip inspect** global configuration command.

---

**Note** The **no ip inspect** 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.

## CBAC Configuration Example

This sample configuration file shows 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:

- Ethernet 0 connects to the internal protected network
- Serial 0 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.
!-----
!
version 11.2
!
! The following three commands should appear in almost every config
!
service password-encryption
service udp-small-servers
no service tcp-small-servers
!
hostname fred-examplecorp-fr
!
boot system flash c1600-fw1600-l
enable secret 5 <elided>
!
username fred 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 3600
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.
! Any traffic that passes the access list will be inspected by CBAC.
! (Traffic blocked by the access list will not be inspected.)
!
interface Ethernet0
description ExampleCorp Ethernet chez fred
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

```

## CBAC Configuration Example

---

```
no ip route-cache
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 ip route-cache
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
no ip route-cache
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 aren't 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
```

```

! Port 22 is SSH... encrypted, RSA-authenticated remote login. Can be used to get to
! field office host from ExampleCorp headquarters.
access-list 111 permit tcp any host 172.19.139.2 eq 22
! 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 igmp any any
! These are the ICMP types actually used...
! administratively-prohibited is useful when you're trying to figure out why
! you can't 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 unreachable 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 unreachable or no unreachable 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

```

