CHAPTER 8

Advanced Security Services, Part II: IOS Firewall Feature Set

As mentioned in the introduction of Chapter 7, "Advanced Security Services, Part I: IPsec," the public Internet is a wonderful extension of the private network, providing global reach to numerous services and vast repositories of data. However, the openness and public nature of the Internet requires you to take precautions. That is, you must protect your private world of users and information from the external swarm of chaos and hackers.

This chapter presents some strategies for defending the perimeter of your private network with two key IOS services: the Firewall feature set and the Intrusion Detection System.

The main topics of this chapter are

- IOS Firewall Fundamentals
- Defending the Perimeter Against Attacks
- How Context-Based Access Control Works
- Configuring CBAC
- Adjusting CBAC Timers and Thresholds
- Enabling Auditing of Sessions
- CBAC with a Demilitarized Zone
- Notes on CBAC Performance
- Configuring Java Applet Blocking for Security
- The IOS Intrusion Detection System

IOS Firewall Fundamentals

The IOS Firewall feature set is an optional, add-on software license for Cisco routers that provides firewall functionality integrated in an IOS router. This is useful for enhancing security at the perimeters of a network—for example, connections to the Internet and links to business partners. Also, because the feature set is IOS-based, it might be a convenient add-on for locations with existing routers or for organizations that need the integration of routing and firewalling in a single platform.

The IOS Firewall feature set provides the following security services:

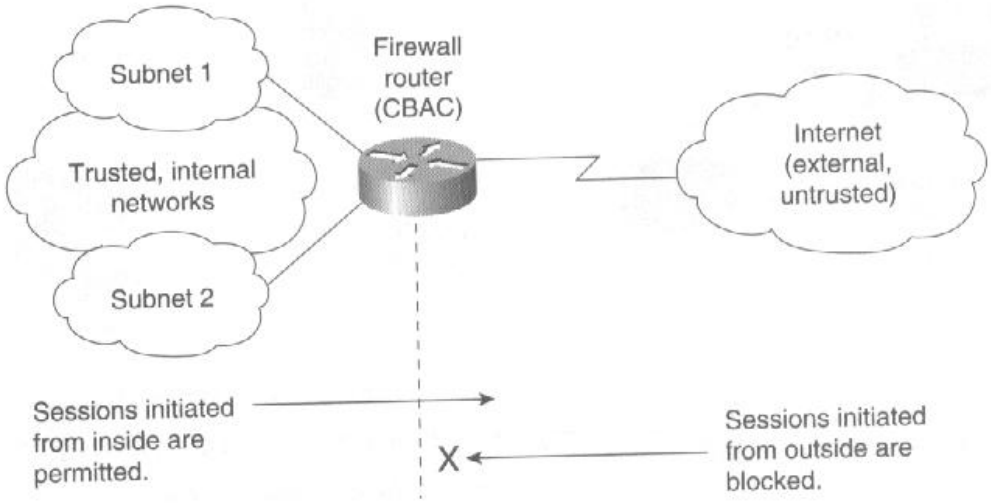
- **Stateful, application-level filtering**—Called context-based access control (CBAC), this service dynamically examines and maintains application-layer protocol information. IOS uses this information to intelligently filter traffic flowing through the firewall router (the router running CBAC).
- **Denial of service detection and prevention**—Denial of service attacks attempt to cripple or disable an IP host by abusing certain vulnerabilities in the host's TCP/IP implementation. CBAC defends against these attacks and reports denial of service activity.
- **Real-time alerts**—CBAC detects certain application attacks and notifies you when they occur. Using the router's syslog reporting service (or console), you can collect and monitor these attacks as part of your security strategy.
- **Auditing of sessions (optional)**—You can enable transactional logging of every session that occurs through the firewall. As sessions end, CBAC reports these statistics via syslog or the console.
- **Java applet blocking (optional)**—CBAC can block Java applets from entering your network, based on a user-defined access list of friendly and hostile external sites.
- **Active auditing of the network with the Intrusion Detection System**—An independent service that you can run with or without CBAC.

Defending the Perimeter Against Attacks

The point where your network connects to the Internet (or other untrusted network) marks the end of your internal network, or the network *perimeter*. The perimeter defines the boundary between the generally controlled, stable internal network and the generally uncontrollable, chaotic outside world. At the perimeter, you must prevent unauthorized access to the internal network from the outside while allowing your internal users access to services on the Internet. This is the fundamental defense provided by a firewall: It allows internal users access to external services on the Internet and, at the same time, prevents external users from accessing the internal network. This defense is provided by CBAC and is depicted in Figure 8-1.

External users might need to access some of your internal resources. Web servers, for example, provide the public with information published by your organization. Other servers, such as Domain Name System (DNS), Simple Mail Transfer Protocol (SMTP), and File Transfer Protocol (FTP), must also be accessible from the outside. For these servers, the firewall must allow just enough access for the services to work and must defend the servers against denial of service attacks.

Figure 8-1 *CBAC (IOS Firewall Feature Set) Allows Outbound Internet Sessions but Blocks Incoming Sessions*



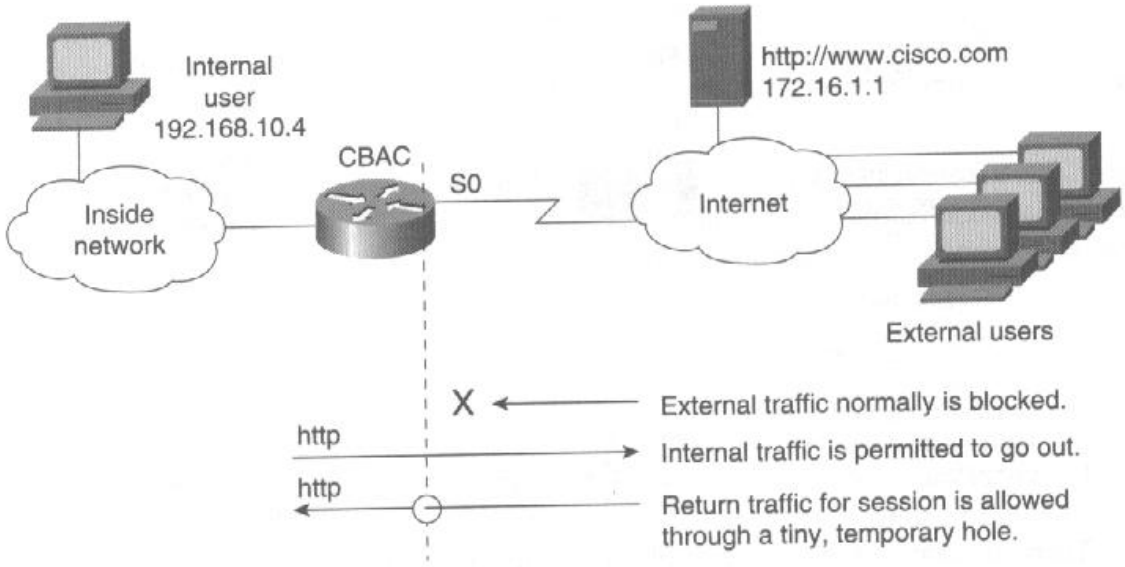
There are many kinds of denial of service attacks, and new attacks are invented on a regular basis. Typical denial of service attacks attempt to cripple or disable an IP host by flooding it with illegal or unusual traffic generated by a custom-made hacker program. That is, the attack *denies* others from using the services on the host for legitimate purposes. Some of the most common denial of service attacks are Teardrop, Land, SYN Flood, and FIN Flood. A firewall defends against these attacks by monitoring traffic to the servers, detecting certain characteristics of the attacks, and intercepting the attacks before they get a chance to overload the servers. Such a defense is provided by CBAC.

NOTE The title *hacker* is traditionally given to a skillful software programmer—someone who masterfully hacks code. Over the years, however, it has become a derogatory name for troublemakers who break into computer systems and networks (and do other menacing activities).

How Context-Based Access Control Works

When an internal user initiates a session (for example, a Web page retrieval) to the Internet, CBAC allows return traffic for that session to pass through access lists that normally prevent external traffic from entering the internal network. CBAC allows the return traffic through to the internal network by dynamically adding temporary access list rules to existing access lists. These temporary CBAC rules are very specific and only permit traffic associated with the user's session—without the rules, the user's application session would fail because its return traffic would be blocked. You can think of the CBAC rules as "tiny holes" in an access list that normally blocks outside traffic. The concept is depicted in Figure 8-2.

Figure 8-2 *CBAC Creates Temporary Access List Rules*



As shown in Figure 8-2, externally sourced packets not associated with the user's session (or other users' sessions also managed by CBAC) are blocked by the access list—this prevents anyone from initiating a session to the internal network from the outside.

As a security precaution, CBAC removes the temporary rules from access lists after the user's session ends, thereby sealing the tiny holes that were needed for the user's session.

Configuring CBAC

These are the basic steps for configuring CBAC:

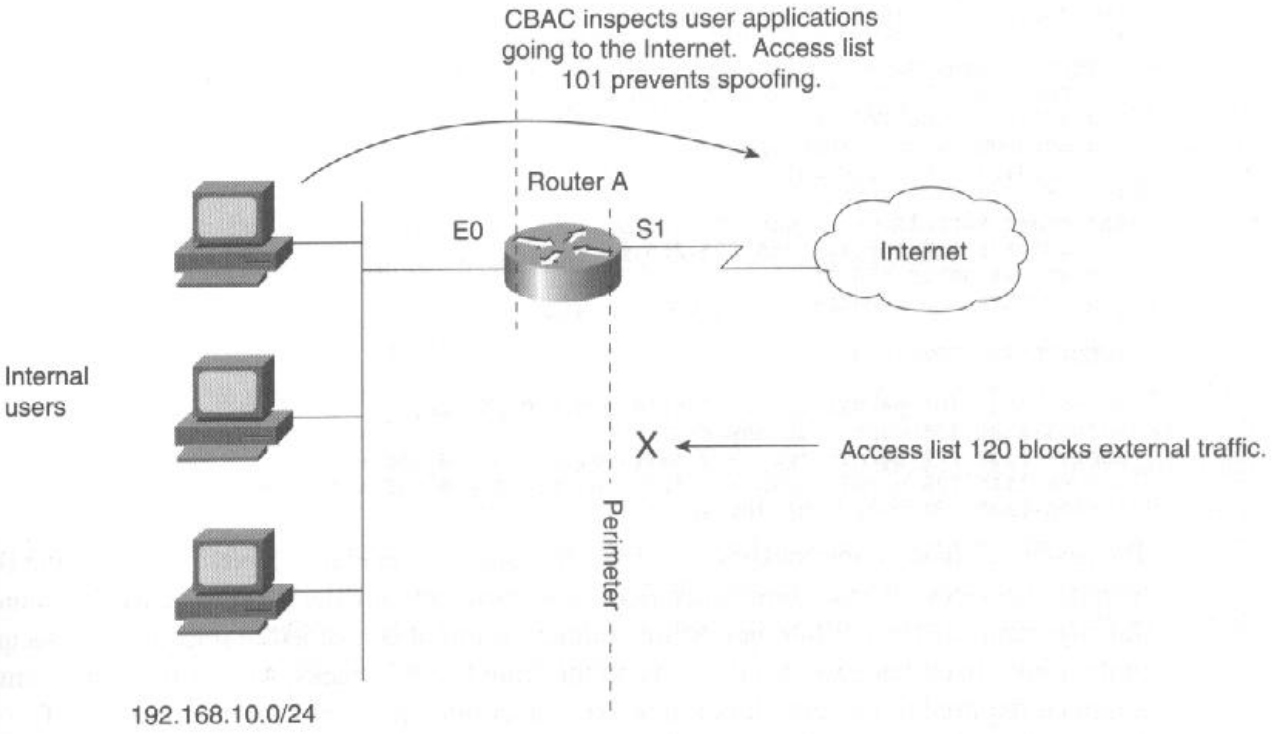
1. Configure and apply extended access lists to block external traffic from entering the internal network.
2. Define a CBAC inspection rule that tells the router which applications require CBAC to function properly.
3. Apply the inspection rule to an interface.

The following sections cover these steps in more detail, starting with a simple two-port firewall example.

CBAC Example: A Basic Two-Port Firewall

Consider the scenario depicted in Figure 8-3 with a basic two-port firewall router.

Figure 8-3 *A Two-Port Firewall Example*



The network in Figure 8-3 is a straightforward application of CBAC. Router A is configured with CBAC and inspects application sessions that originate from the inside and are destined to the Internet. An inbound access list is applied to Router A's Serial1 interface that faces the public Internet; the access list guards against unauthorized entry from the Internet to the internal network. Finally, CBAC allows return traffic for internal user applications (Web, Telnet, FTP, and so on) through the access list—the return traffic must be permitted for the applications to function properly.

This two-port configuration is good for explaining basic CBAC configuration. This scenario might be appropriate for a home office or an organization that does not directly offer Web, FTP, DNS, or SMTP servers to the public. Such organizations might make their public servers available through an ISP that hosts the servers at an off-site location. See "CBAC with a Demilitarized Zone" later in this chapter for a CBAC example that makes servers available from the outside.

The following is the configuration for the firewall router, Router A (for brevity, only the lines relevant to security and CBAC are shown):

```
no service tcp-small-servers
no service udp-small-servers
service password-encryption
enable secret <my-password>
no ip source-route
no cdp run
!
```

continues

```
hostname RTA
!
ip inspect name INSPECT-RULE1 tcp
ip inspect name INSPECT-RULE1 udp
!
interface Ethernet0
 ip address 192.168.10.1 255.255.255.0
 ip access-group 101 in
 ip inspect INSPECT-RULE1 in
 no ip directed-broadcast
!
interface Serial1
 ip address 192.168.1.2 255.255.255.0
 ip access-group 120 in
 no ip directed-broadcast
!
logging 192.168.10.3
!
access-list 101 permit ip 192.168.10.0 0.0.0.255 any
access-list 101 deny   ip any any
access-list 120 permit icmp any 192.168.10.0 0.0.0.255 echo
access-list 120 permit icmp any 192.168.10.0 0.0.0.255 echo-reply
access-list 120 deny   ip any any
```

The first block of six lines consists of security measures covered in Chapter 6, "Deploying Basic Security Services." These commands increase security and are desirable because this router is directly connected to the Internet. More commands might be needed to reinforce the security of the router itself because this router is on the front line of attacks. Access lists, for example, might be required to prevent all possible external communication with the router itself. Passwords on the vty interfaces are highly recommended (see Chapter 6).

The command **ip inspect name INSPECT-RULE1 tcp** does two things:

- It creates a CBAC *inspection rule* called **INSPECT-RULE1**. An inspection rule defines the applications (HTTP, FTP, H.323, SQL*Net, and so on) that CBAC observes and manages. CBAC creates the temporary access list rules mentioned earlier for the applications listed in the inspection rule.
- It tells CBAC to inspect generic TCP applications as designated by the **tcp** keyword. Generic TCP applications include Web (http), Telnet, Usenet news (NNTP), Time (NTP), passive-mode FTP, and other applications that do not carry application-specific address or port information within the TCP packet. In other words, return packets for generic TCP applications have the same address and port number as the user-sourced packets that exited the network (with source and destination fields reversed). See "Configuring CBAC Inspection of Other Applications" later in this chapter for configuring applications that require application-layer CBAC inspection.

The command **ip inspect name INSPECT-RULE1 udp** adds to the rule **INSPECT-RULE1** and enables generic UDP application inspection. As with generic TCP inspection, UDP inspection supports applications that return packets over the same address and port numbers as exiting packets (with source and destination fields reversed). DNS is an example of a generic UDP application.

The next block of commands pertains to the router's Ethernet0 interface that connects to the internal LAN:

- The command **ip access-group 101 in** applies access list 101 to the interface and specifies the inbound direction. Access list 101 is an anti-spoofing access list that ensures all packets coming from the LAN (entering Ethernet0) have a valid source address from the 192.168.10.0/24 subnet. This is a good security policy: It prevents mischievous *internal* users (or attackers that have gained access to your network) from launching spoof attacks to the Internet and thereby causing potential legal problems for your organization. See Chapter 6 for more information on access lists and spoofing.
- The command **ip inspect INSPECT-RULE1 in** applies the previously configured inspection rule **INSPECT-RULE1** to the interface and specifies the inbound direction. The inspection rule does not do anything until you apply it to an interface with this command. The keyword **in** is very important and defines the *direction of inspection* for the applications defined in INSPECT-RULE1. It tells CBAC to inspect applications that are initiated from the internal LAN and *enter* Ethernet0. Alternatively, you could apply INSPECT-RULE1 to Serial1 in the *outbound* direction (keyword **out**) and accomplish the same thing because user-initiated packets flow out of Serial1. You can think of the direction of inspection as the direction of session initiation: Sessions originate at the internal users, enter Ethernet0, and exit out Serial1.

NOTE For the technically inclined, direction of inspection is the path of the first TCP SYN packet (or the first UDP packet). From the router's point of view, it expects the first packet to arrive *inbound* on Ethernet0; thus, the keyword **in** is needed in this example.

- The command **no ip directed-broadcast** is a security measure and protects the internal LAN against attacks that generate floods with IP broadcasts. See Chapter 6 for more information.

The next block of commands pertains to the router's Serial1 interface that connects to the Internet:

- The command **ip access-group 120 in** applies access list 120 to the interface and specifies the inbound direction. Access list 120 is the heart of the protection provided by the firewall router. It blocks all traffic from the Internet except pings (ICMP echo) for troubleshooting purposes. CBAC dynamically adds temporary rules to this access list ("pokes holes" in the list) to permit return traffic of legitimate, internally initiated sessions.

NOTE If ping flooding attacks are a concern, they can be denied—common ping attacks attempt to consume your bandwidth with floods of ping packets. Alternatively, if more types of packets need to be permitted, the access list can permit them. Other external packets you might consider permitting are ssh (secure shell, port 22) and the following ICMP messages: unreachable, traceroute, time exceeded, administratively prohibited, and packet-too-big (path MTU discovery).

- The command **no ip directed-broadcast** blocks outgoing IP broadcasts to the Internet. In the event an attacker gains access to your network and attempts to use it as a launch pad for attacking other networks, this command prevents the router from sending IP broadcasts to the Internet.

The command **logging 192.168.10.3** tells the router to send IOS system and error messages to a syslog server whose address is 192.168.10.3. This is useful for remotely monitoring CBAC alerts and all other router messages. See Appendix E, "A Crash Course in Cisco IOS," for more information on syslog.

The remaining commands configure access lists 101 and 120. Access list 101 is applied to Ethernet0 in the inbound direction:

- The rule **access-list 101 permit ip 192.168.10.0 0.0.0.255 any** permits source addresses from the internal subnet 192.168.10.0/24 only.
- The rule **access-list 101 deny ip any any** is not really necessary because it is the same as the invisible rule at the end of every IP access list (see Chapter 6). However, it is explicitly configured to make more visible and more obvious the denial of all packets not matching the preceding rule.

Access list 120 is applied to Serial1 in the inbound direction:

- The rules **access-list 120 permit icmp any 192.168.10.0 0.0.0.255 echo** and **access-list 120 permit icmp any 192.168.10.0 0.0.0.255 echo-reply** permit ping packets from the outside to the internal LAN. This is for troubleshooting purposes and might not be desirable if ping attacks are a concern.
- The rule **access-list 120 deny ip any any** makes obvious the denial of all other packets.

Validating CBAC Configuration

To validate that CBAC is working properly, issue the **show ip inspect all** enable mode command. The following is an output from the previous two-port firewall example:

```
RTA#sh ip ins 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 INSPECT-RULE1
    tcp timeout 3600
    udp timeout 30

Interface Configuration
 Interface Ethernet0
  Inbound inspection rule is INSPECT-RULE1
    tcp timeout 3600
    udp timeout 30
  Outgoing inspection rule is not set
  Inbound access list is 101
  Outgoing access list is not set

Established Sessions
 Session B0893C (192.168.10.4:1148)=>(172.16.224.209:23) tcp SIS_OPEN
 Session AF7C4C (192.168.10.4:1147)=>(172.16.10.72:53) udp SIS_OPEN
```

The preceding output of **show ip inspect all** displays

- The current CBAC settings, such as various timers and thresholds covered in "Adjusting CBAC Timers and Thresholds" later in this chapter.
- The names of inspection rules and the applications that are defined for each rule. In this example, the router has just one rule, called **INSPECT-RULE1**, and this rule is inspecting generic TCP and UDP applications only.
- The interfaces configured with CBAC, along with inspection rule names and access list numbers.
- The active sessions that CBAC is inspecting and managing. **Session B0893C** displays a unique number that the router uses to identify the session. **(192.168.10.4:1148)=>** is the internal address and port number that initiated the session. **(172.16.224.209:23)** is the external destination address and port number. **tcp** indicates this is a generic TCP application, and **SIS_OPEN** means the connection status is open and active. Other status messages you'll typically see are **SIS_OPENING** (connection half-open) and **SIS_CLOSING** (connection closing).

To verify that CBAC is indeed adding temporary rules to access lists that block external traffic, issue the **show access-lists** command. Recall that, in the previous two-port firewall example, access list 120 is filtering inbound on Serial1.

```
RTA#sh access-l
Extended IP access list 101
    permit ip 192.168.10.0 0.0.0.255 any (31970 matches)
    deny ip any any (4 matches)
Extended IP access list 120
    permit tcp host 172.16.224.209 eq telnet host 192.168.10.4 eq 1148 (19 matches)
```

continues

```
permit udp host 172.16.10.72 eq domain host 192.168.10.4 eq 1147 (1 match)
permit icmp any 192.168.10.0 0.0.0.255 echo
permit icmp any 192.168.10.0 0.0.0.255 echo-reply (4 matches)
deny ip any any (2497 matches)
```

The first two rules in access list 120 were dynamically created by CBAC to permit return traffic for two very specific internally initiated applications: a Telnet session and a DNS query (keyword **domain**). Notice that CBAC inspection pinpoints the specific port numbers for both the source and destination and allows only these packets to pass through the interface.

NOTE When the router has multiple access lists that block return traffic, CBAC adds temporary rules to the lists that require them to make the application work successfully.

After the Telnet session terminates, CBAC promptly removes the temporary access list rules. This stateful operation is verified by issuing **show access-lists** again:

```
RTA#sh access-l
Extended IP access list 101
    permit ip 192.168.10.0 0.0.0.255 any (32043 matches)
    deny ip any any (4 matches)
Extended IP access list 120
    permit icmp any 192.168.10.0 0.0.0.255 echo
    permit icmp any 192.168.10.0 0.0.0.255 echo-reply (4 matches)
    deny ip any any (2497 matches)
```

The preceding output shows that the temporary rules no longer exist in access list 120 after the Telnet session has ended. List 120 is back to its original rule list.

NOTE Because of the connectionless nature of UDP applications, CBAC has no definitive way of determining the end of a UDP session. Therefore, CBAC approximates the end of a UDP session by expiring a configurable period of inactivity before declaring the session over. See also "Adjusting CBAC Session Timers," later in this chapter.

Configuring CBAC Inspection of Other Applications

In addition to generic TCP and UDP applications, CBAC supports some popular applications that require inspection of application-layer protocols to function properly. To get a current list of supported application-layer protocols, check the Cisco documentation or boot up a recent release of IOS with CBAC and look at the context-sensitive help for the **ip inspect name** global config command:

```
RTA#conf t
Enter configuration commands, one per line. End with CNTL/Z.
RTA(config)#ip inspect name MY-RULE ?
  cuseeme       CUSeeMe Protocol
  fragment      IP fragment inspection
  ftp           File Transfer Protocol
  h323          H.323 Protocol (e.g, MS NetMeeting, Intel Video Phone)
  http          HTTP Protocol
  rcmd          R commands (r-exec, r-login, r-sh)
  realaudio     Real Audio Protocol
  rpc           Remote Procedure Call Protocol
  smtp          Simple Mail Transfer Protocol
  sqlnet        SQL Net Protocol
  streamworks   StreamWorks Protocol
  tcp           Transmission Control Protocol
  tftp          TFTP Protocol
  udp           User Datagram Protocol
  vdolive       VDOLive Protocol
  <cr>
```

To add support for H.323 and FTP applications in the previous two-port firewall example, add the necessary lines to the inspection rule **INSPECT-RULE1**:

```
hostname RTA
!
ip inspect name INSPECT-RULE1 tcp
ip inspect name INSPECT-RULE1 udp
ip inspect name INSPECT-RULE1 ftp
ip inspect name INSPECT-RULE1 h323
```

The commands **ip inspect name INSPECT-RULE1 ftp** and **ip inspect name INSPECT-RULE1 h323** enable CBAC support for FTP and H.323 applications, respectively. Internal users can now successfully communicate through the firewall to the Internet with these applications.

NOTE The command **ip inspect name *<rule-name>* ftp** permits the use of regular FTP sessions, but most FTP clients and servers support an option called *passive mode* that works fine with CBAC's generic TCP inspection.

CBAC provides some additional security benefits for application-protocols such as SMTP by watching for suspicious application-layer commands. Also, Cisco routinely updates CBAC with new defenses; however, as mentioned in Chapters 6 and 7, there is no such thing as a hacker-proof network. A service such as CBAC can enhance security and defend against common attacks, but cannot defend against yet-to-be-invented attacks.

Keep Your Security Systems Current

It is highly recommended that you maintain a constant process of staying current with security advisories, implementation of defenses, and auditing of the effectiveness of your security systems. Here are some security resources that will get you started:

- **Cisco security advisories mailing list**—To subscribe to the advisories mailing list, send a message to majordomo@cisco.com. In the body of the message, include the single line "info cust-security-announce" (without the quotation marks).
- **Cisco security discussion mailing list**—To subscribe to the discussion mailing list, send a message to majordomo@cisco.com. In the body of the message, include the single line "info cust-security-discuss" (without the quotation marks).
- **CERT®** —http://www.cert.org
- **CIAC (Computer Incident Advisory Capability)**—http://ciac.llnl.gov
- **Bugtraq**—This e-mail discussion list reports security holes in operating systems and applications. To subscribe, send a message to listserv@securityfocus.com. In the body of the message include the single line "subscribe bugtraq" (without the quotation marks). Go to http://www.securityfocus.com for archives of the list and other security information.
- **FIRST (Forum of Incident Response and Security Teams) mailing list**—Another good security mailing list. To subscribe, send a message to first-majordomo@first.org. In the body of the message, include the single line "subscribe first-info" (without the quotation marks).
- **RFC 2196, "Site Security Handbook"**

Adjusting CBAC Timers and Thresholds

CBAC maintains several timers and thresholds that define how long the temporary access list rules are active and how aggressively CBAC combats denial of service attacks. The following sections cover CBAC's adjustable parameters in detail.

Adjusting CBAC Session Timers

To change the default timers associated with CBAC, issue one or more of the following global config commands (you need to issue these commands only if you want to change the built-in CBAC defaults):

- **ip inspect tcp synwait-time *<seconds>*** configures the number of seconds CBAC waits for a half-open TCP session to be established before dropping it. The default is 30 seconds. In most cases, you will not have to modify this value.

NOTE A half-open session is a session that has not yet completed TCP's *three-way handshake* process. A complete handshake between two hosts consists of a SYN (synchronization) packet from the initiator, followed by a SYN+ACK (SYN plus acknowledgement) packet from the destination, and then a final ACK packet from the initiator.

- **ip inspect tcp finwait-time *<seconds>*** configures the number of seconds CBAC continues to manage a TCP session after observing a TCP FIN-exchange (a FIN-exchange between two hosts terminates the TCP session). The default is 5 seconds. In most cases, you will not have to modify this value.
- **ip inspect tcp idle-time *<seconds>*** configures the number of seconds a TCP session may remain idle before CBAC terminates the session and removes it from its state table. The default is 3,600 seconds (1 hour). It might be desirable to lower this value and close CBAC's temporary holes for inactive sessions sooner (to increase security and reduce the number of stale CBAC rules in an access list). As a possible downside of lowering this timer, your users might complain that their sessions get disconnected after they leave their computers for a while. This idle timer can be overridden on a per-application basis by defining timeout values in an inspection rule (see the following section, "Overriding Global Timers with Inspection Rules").
- **ip inspect udp idle-time *<seconds>*** configures the number of seconds a UDP session may remain idle before CBAC declares the session over. Because of the connectionless nature of UDP, CBAC approximates the end of a UDP session by expiring this period of inactivity. The default is 30 seconds. You might have to increase this value slightly if CBAC is undesirably closing UDP applications too aggressively. This idle timer can be overridden on a per-application basis by defining timeout values in an inspection rule (see the following section, "Overriding Global Timers with Inspection Rules").
- **ip inspect dns-timeout *<seconds>*** configures the number of seconds a DNS session may remain idle before CBAC declares the session over and removes the temporary access list rule. Because DNS queries are quick and short-lived, CBAC maintains a short idle timer just to guard against attacks on DNS servers. The default is 5 seconds. You should not have to modify this value.

Overriding Global Timers with Inspection Rules

You can override the global TCP and UDP idle timers for certain applications by defining timeout values as you configure inspection rules with the **ip inspect name** command.

The following example configures inspection rule **INSPECT-RULE1** with SQL*Net support and defines a specific idle timeout for SQL*Net sessions only:

```
RTA(config)#ip inspect name INSPECT-RULE1 sqlnet timeout 900
```

This command tells CBAC to close SQL*Net sessions after 900 seconds of inactivity. Other applications may have different timeouts or may inherit the global values set by **ip inspect tcp idle-time** and **ip inspect udp idle-time** (see the preceding section, "Adjusting CBAC Session Timers").

Adjusting CBAC Denial of Service Thresholds

CBAC detects and prevents common denial of service attacks as part of its inspection process. As mentioned earlier, denial of service attacks attempt to cripple or disable a server by flooding it with illegal or unusual traffic.

You can tune the default thresholds that govern how aggressively CBAC combats these attacks with the following global config commands (you need to issue these commands only if you want to change the built-in CBAC defaults):

- **ip inspect max-incomplete high *<number>*** configures the number of half-open sessions that will trigger CBAC to start deleting half-open sessions. If you do not issue this command, the default number is 500. A high number of half-open sessions is suspicious and might mean a denial of service attack is happening (see "Adjusting CBAC Session Timers," earlier in this chapter). When the number of half-open sessions reaches the threshold defined by this command, CBAC deletes half-open sessions until the number of half-open sessions falls below the threshold defined by the following command, **ip inspect max-incomplete low**. CBAC does this so that the number of half-open sessions is limited and manageable and so that existing half-open sessions are deleted to accommodate new, and possibly legitimate, session requests.

NOTE For TCP applications, half-open means the session has not yet completed the TCP three-way handshake (see "Adjusting CBAC Session Timers," earlier in this chapter). For UDP applications, half-open means the session was initiated but hasn't experienced any return traffic.

- **ip inspect max-incomplete low *<number>*** configures the number of half-open sessions that causes CBAC to stop deleting half-open sessions. This command works in conjunction with the preceding command, **ip inspect max-incomplete high**. After CBAC begins deleting half-open sessions as a result of reaching the upper threshold defined by

ip inspect max-incomplete high, it stops deleting when the number of half-open sessions reaches the number defined by **ip inspect max-incomplete low**. The default for this lower threshold is 400 half-open sessions.

- **ip inspect one-minute high *<number>*** configures the upper threshold for the number of new half-open sessions per minute. This measures the *rate* of new half-open sessions per minute instead of the absolute number of half-open sessions that exist in the router. It's the same idea as **ip inspect max-incomplete high** but takes into account the added dimension of time. By default the number is 500, which means CBAC will start deleting half-open sessions when the rate of new half-open sessions reaches 500 per minute.
- **ip inspect one-minute low *<number>*** configures the lower threshold for the number of new half-open sessions per minute. This threshold, like the **ip inspect max-incomplete low** command, tells CBAC when to stop deleting half-open sessions—except this command takes into account the rate of new half-open sessions per minute. After CBAC begins deleting half-open sessions as a result of reaching the upper threshold defined by **ip inspect one-minute high**, it stops deleting when the rate of half-open sessions reaches the number defined by **ip inspect one-minute low**. The default for this lower threshold is 400 half-open sessions per minute.
- **ip inspect tcp max-incomplete host *<number>*** configures the number of half-open TCP sessions that will trigger CBAC to start deleting half-open sessions *per destination host*. This protects hosts by limiting the number of half-open TCP sessions any one host can have (of course, CBAC must be inspecting the sessions to the host). The default limit is 50 half-open TCP sessions. When this limit is reached, CBAC deletes an old half-open session for every new session that is passed to the host. Optionally, you can specify a *block time* that temporarily shields hosts from new session requests after the limit is reached. For example, the command **ip inspect tcp max-incomplete host 50 block-time 2** tells CBAC to drop all new session requests to a host for 2 minutes after the host has 50 half-open sessions. The block time protects hosts from denial of service storms and is recommended when you need to protect against high-rate, high-bandwidth denial of service attacks.
- **ip inspect name *<rule-name>* fragment** adds a feature to your inspection rule that protects against IP fragmentation attacks. These attacks attempt to consume and freeze server resources by sending packets that are illegally fragmented or intentionally incomplete. With this command, CBAC detects and prevents certain fragmentation attacks within the traffic it inspects. Consult the IOS command reference for specifics and tunable parameters.

NOTE Just about every workstation, server, and operating system vendor has issued software updates to defend against common denial of service attacks. It is highly recommended that you check with the manufacturers of your host systems and implement these updates. The security Web sites and mailing lists mentioned in "Keep Your Security Systems Current" also have information on software updates.

Enabling Auditing of Sessions

An option is available in CBAC that enables you to audit the sessions inspected by CBAC. Issue the **ip inspect audit-trail** global config command to do this:

```
RTA(config)#ip inspect audit-trail
```

Now, when a session ends you will receive a system message like this:

```
%FW-6-SESS_AUDIT_TRAIL: tcp session initiator (192.168.10.4:1246) sent 109 bytes -
- responder (172.17.24.20:23) sent 22806 bytes
```

If you want more granular control of audit trail messages, use the keyword **audit-trail** when you define the applications in your inspection rule:

```
RTA(config)#no ip inspect audit-trail
RTA(config)#ip inspect name INSPECT-RULE1 tcp
RTA(config)#ip inspect name INSPECT-RULE1 udp
RTA(config)#ip inspect name INSPECT-RULE1 ftp
RTA(config)#ip inspect name INSPECT-RULE1 h323 audit-trail on
```

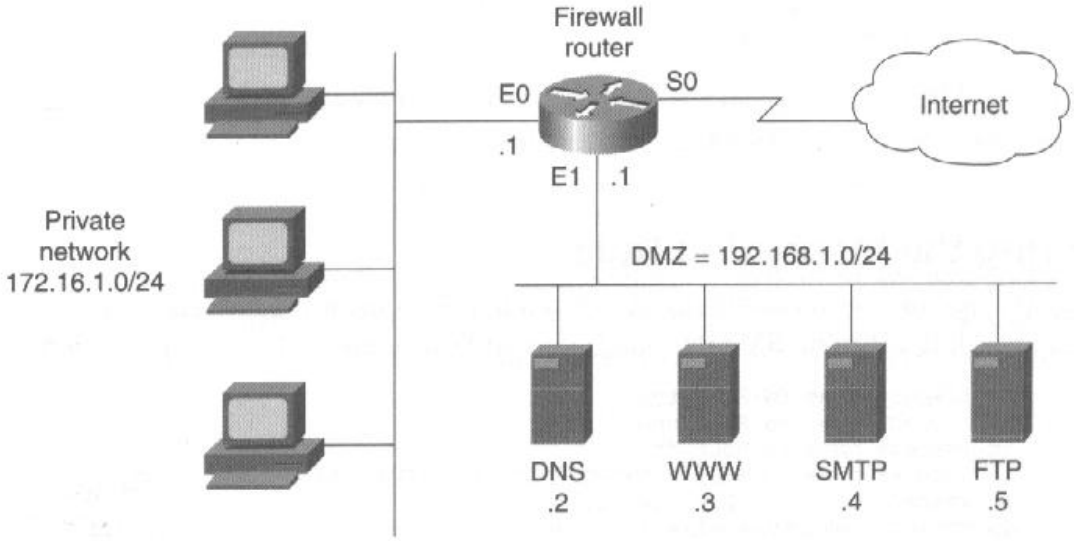
where **no ip inspect audit-trail** turns off auditing by default and **ip inspect name INSPECT-RULE1 h323 audit-trail on** enables auditing for H.323 applications only. No audit messages will be displayed for the other applications: generic TCP, generic UDP, and FTP.

NOTE Auditing on a per-application basis with **ip inspect name audit-trail on** is not available in some older releases of the IOS Firewall feature set.

CBAC with a Demilitarized Zone

A simple two-port firewall was presented earlier in this chapter to explain the basics of CBAC configuration (see "CBAC Example: A Basic Two-Port Firewall"). Now consider a more complex CBAC configuration that includes a third network port to connect a *demilitarized zone (DMZ)*—a publicly accessible network that provides public Web, SMTP (e-mail), DNS, and other servers. Suppose a company, Widget, Inc., has the firewall and DMZ topology depicted in Figure 8-4.

Figure 8-4 *A More Complex CBAC Example: Firewall and DMZ*



Widget, Inc., has the following security requirements:

- No traffic is permitted from the Internet to the private network except return traffic for sessions initiated by the internal users.
- Traffic is permitted from the Internet to the DMZ for accessing public DNS, Web, e-mail, and FTP servers. This traffic is inspected for denial of service attacks.
- Traffic is permitted from the private network to the Internet and to the DMZ.
- ICMP echo (ping) traffic is permitted to the private network and DMZ for troubleshooting network reachability.
- No traffic is permitted from the DMZ to the Internet except return traffic for sessions initiated by users on the Internet.

The following sections walk through the IOS configuration of the firewall router and demonstrate how CBAC is used to meet the security objective.

Basic Security Commands for the Firewall Router

The following portion of the firewall router's configuration implements basic IOS security principles (these are covered in Chapter 6):

```
no service tcp-small-servers
no service udp-small-servers
service password-encryption
enable secret <my-password>
no ip source-route
no cdp run
!
```

continues

```
interface Ethernet0
 no ip directed-broadcast
!
interface Ethernet1
 no ip directed-broadcast
!
line 0 6
 login
 password <my-password>
```

Configuring the Inspection Rule

Widget, Inc., next configures its inspection rule to support generic TCP, generic UDP, FTP, H.323, RealAudio, SMTP (e-mail), StreamWorks, and VDOLive applications:

```
ip inspect name FW-RULE tcp
ip inspect name FW-RULE udp
ip inspect name FW-RULE ftp
ip inspect name FW-RULE h323
ip inspect name FW-RULE realaudio
ip inspect name FW-RULE smtp
ip inspect name FW-RULE streamworks
ip inspect name FW-RULE vdolive
ip inspect name FW-RULE fragment
```

The last line in the preceding list, **ip inspect name FW-RULE fragment**, enables protection against fragmentation-based denial of service attacks. See "Adjusting CBAC Denial of Service Thresholds," earlier in this chapter.

Configuring the Private Network Interface

The following portion of the Widget, Inc., configuration implements access control and CBAC inspection for the private network interface Ethernet0:

```
interface Ethernet0
 description Interface to the Private Network
 ip address 172.16.1.1 255.255.255.0
 ip access-group 101 in
 ip access-group 121 out
 ip inspect FW-RULE in
!
access-list 101 permit ip 172.16.1.0 0.0.0.255 any
access-list 101 deny ip any any
access-list 121 permit icmp any any echo
access-list 121 permit icmp any any echo-reply
access-list 121 deny ip any any
```

The command **ip access-group 101 in** applies an inbound anti-spoofing access list to Ethernet0, thus ensuring that packets entering the router from the private network have legitimate source addresses (addresses from subnet 172.16.1.0/24). See Chapter 6 for more information on anti-spoofing access lists.

The command **ip access-group 121 out** applies an outbound access list that blocks all traffic to the private network except ICMP echo (ping) packets. CBAC will add temporary access list rules to this access list to permit return traffic for sessions that are initiated by the internal clients.

The command **ip inspect FW-RULE in** applies the inspection rule **FW-RULE** and specifies the inbound direction. All application sessions entering the router through this interface will be inspected by CBAC. Then, CBAC will create the necessary access list rules to permit return traffic for those sessions.

The remaining lines are the global config commands that define the anti-spoofing access list 101 and the firewall access list 121. As mentioned earlier, the command **access-list <#> deny ip any any** is not necessary but makes the meaning of the access list more obvious.

Configuring the DMZ Network Interface

The next portion of the Widget, Inc., configuration implements access control on the DMZ network interface Ethernet1:

```
interface Ethernet1
 description Interface to the DMZ
 ip address 192.168.1.1 255.255.255.0
 ip access-group 102 in
 ip access-group 122 out
```

Access list 102 is an anti-spoofing access list that ensures all packets entering the router from the DMZ network have legitimate source addresses (addresses from subnet 192.168.1.0/24). Here are the lines that define access list 102:

```
access-list 102 permit ip 192.168.1.0 0.0.0.255 any
access-list 102 deny ip any any
```

Access list 122 controls the traffic that can enter the DMZ (exit Ethernet1). First, the list must allow Internet users access to the DNS, Web, SMTP, and FTP servers on the DMZ. This is defined by the following access list rules:

```
access-list 122 permit udp any host 192.168.1.2 eq domain
access-list 122 permit tcp any host 192.168.1.3 eq www
access-list 122 permit tcp any host 192.168.1.4 eq smtp
access-list 122 permit tcp any host 192.168.1.5 eq ftp
```

TIP To allow DNS zone transfers, add a rule to permit DNS over TCP. Opening DNS servers to zone transfers is sometimes considered a security risk: The information stored in your DNS server could be useful to an attacker.

Next, list 122 must permit traffic from the private network to the DMZ. This was one of the requirements of Widget, Inc., and allows the internal users to use the public services on the DMZ. The following rule accomplishes this:

```
access-list 122 permit ip 172.16.1.0 0.0.0.255 192.168.1.0 0.0.0.255
```

The list then permits ICMP echo traffic to the DMZ:

```
access-list 122 permit icmp any any echo
access-list 122 permit icmp any any echo-reply
```

Finally, the list denies all other traffic:

```
access-list 122 deny ip any any
```

Configuring the Internet Interface

The final portion of the Widget, Inc., configuration implements access control and CBAC inspection on the Internet interface Serial0:

```
interface Serial0
 description Interface to the ISP
 ip unnumbered ethernet 1
 ip access-group 103 in
 ip access-group 123 out
 ip inspect FW-RULE in
!
access-list 103 deny ip 172.16.0.0 0.0.255.255 any
access-list 103 deny ip 192.168.1.0 0.0.0.255 any
access-list 103 deny ip 127.0.0.0 0.255.255.255 any
access-list 103 permit ip any any
access-list 123 deny ip 192.168.1.0 0.0.0.255 any
access-list 123 permit ip any any
```

The command **ip unnumbered ethernet 1** is covered in Chapter 1, "Managing Your IP Address Space." It configures the interface without a subnet assignment.

The command **ip access-group 103 in** applies an inbound anti-spoofing access list to Serial0 to ensure that packets entering the router from the Internet *do not* have a source address from the Widget, Inc., address space. This protects against spoofing attacks originating from the Internet. See Chapter 6 for more information about spoofing attacks.

The command **ip access-group 123 out** applies an outbound access list that permits all traffic to the Internet except packets from the DMZ. In the event an attacker gains access to the DMZ, this access list prevents the attacker from using the DMZ as a launch pad for attacks to other parts of the Internet.

The command **ip inspect FW-RULE in** applies the inspection rule **FW-RULE** and specifies the inbound direction. The command tells the router to inspect all sessions entering the router from the Internet. With this command, CBAC detects and prevents denial of service attacks

coming from the Internet and destined to the servers on the DMZ. Widget, Inc., can optionally tune CBAC's denial of service parameters to suit its taste (see "Adjusting CBAC Denial of Service Thresholds," earlier in this chapter). CBAC also adds temporary access list rules to outbound access list 123 so traffic from the DMZ is permitted to the Internet if it belongs to an inspected session. This is a reverse logic of the inspection occurring on the private interface Ethernet0. On this interface (Serial0), the sessions are initiated from *external* users on the Internet and the "return traffic" is traffic from the DMZ to the Internet.

Notes on CBAC Performance

Sizing CBAC performance and selecting the router for your network perimeter depends on many factors: for example, the bandwidth of the WAN link you need to support, the average packet size of your traffic, and the IOS services you need to run on the router. Although there are no hard and fast rules, Table 8-1 offers some general guidelines—you might find that your particular environment is more or less forgiving. These guidelines assume the firewall router is dedicated to CBAC processing.

Table 8-1 *General Guidelines for CBAC Routers*

Router Platform	Typical CBAC Capability
Low-end (Cisco 1600, 2500)	Up to T1 (1.544 Mbps)
Mid-range (Cisco 2600, 3600)	Up to 5 Mbps
High-end (Cisco 7200)	Up to T3 (45 Mbps)

NOTE Cisco also manufactures a dedicated firewall appliance called the PIX Firewall, suited for securing networks at higher bandwidth rates. The PIX also supports some unique features not found in the IOS Firewall. Check with Cisco for the most recent performance enhancements and features.

Configuring Java Applet Blocking for Security

You can configure CBAC to inspect Web (HTTP) traffic for Java applets and prevent some or all applets from entering your private network. Blocking of the applets is based on a user-defined list of Web sites—that is, you can define a Web site as hostile and block all applets from that site. This might be useful for protecting internal users from sites known to have malicious Java applets.

The following example configuration blocks Java applets coming from two sites, 172.16.206.33 and 192.168.171.65 (only the lines relevant to configuring the inspection rule are shown):

```
ip inspect name FW-RULE tcp
ip inspect name FW-RULE udp
ip inspect name FW-RULE ftp
ip inspect name FW-RULE h323
ip inspect name FW-RULE realaudio
ip inspect name FW-RULE smtp
ip inspect name FW-RULE streamworks
ip inspect name FW-RULE vdolive
ip inspect name FW-RULE fragment
ip inspect name FW-RULE http java-list 10
!
access-list 10 deny host 172.16.206.33
access-list 10 deny host 192.168.171.65
access-list 10 permit any
```

The command **ip inspect name FW-RULE http java-list 10** adds another line to the inspection rule **FW-RULE** and tells CBAC to block Java applets based on user-defined access list **10**. This access list defines the sites that are deemed to house hostile applets.

The last three lines configure access list 10, which defines (with the **deny** keyword) the addresses known to have hostile applets. This must be a standard, not extended, access list. The last line of access list 10, **access-list 10 permit any**, allows applets from all other sites to pass through the firewall router. Alternatively, you could reverse the logic of the access list and permit applets from approved sites only and deny applets from all other sites.

NOTE CBAC cannot block Java applets that are packaged or "wrapped" in files, such as ZIP and JAR archives. Also, CBAC looks for applets in HTTP port 80 traffic streams and therefore cannot filter applets transferred via other means, such as FTP or HTTP over a nonstandard port.

The IOS Intrusion Detection System

An important part of any good security policy is active auditing of applications on the network and timely detection of hacker activity. It is simply not enough to implement a few perimeter defenses and consider it a job well done. Nor is it wise to sit back and wait for an attacker to foil your defenses and then do something blatant enough to be noticed—many attackers are quiet and hope to roam your network undetected.

Active auditing or *intrusion detection* is a security service offered with the IOS Firewall feature set and acts as a watchdog for your believed-to-be-secure network. This security service provides constant surveillance, detection, termination, and alerting of suspicious network activity.

The IOS Intrusion Detection System (IDS) contains a database of hacking patterns called *signatures*. Each signature describes the characteristics of a particular hacker attack. For

example, hackers who gain access to UNIX hosts often try to download the */etc/passwd* file (a system file that, in some versions of UNIX, contains user login and password information). The IDS signature for this attack could be written in human terms as follows:

> Issuing the string *passwd* during an FTP session is an attack.

The preceding is an attack signature because under normal circumstances, no legitimate user would issue an FTP command containing the word *passwd*. IDS examines the application sessions flowing through the router and looks for matches to this and other signatures. If a hacker types the FTP command **GET /etc/passwd** (or *any* command containing the word **passwd**), IDS immediately detects this, closes the suspicious session, and sends you an alert. You can then review the incident, correct any alterations, and implement appropriate countermeasures.

NOTE The database of signatures is administered by a team of security experts at Cisco. This team follows developments in the hacker community and creates new signatures as they are needed.

The IOS version of IDS is a subset of the technology found in Cisco's standalone intrusion detection product, NetRanger™. NetRanger is a dedicated active audit appliance that provides a full suite of signatures, high performance, and central management. NetRanger also supports a feature called *shunning*: the application of dynamic access lists when an attack is detected to block further hacking. Consult Cisco for more information on NetRanger.

You can deploy the IOS version of IDS standalone or in conjunction with a comprehensive NetRanger system. The following sections describe how to configure IDS on IOS router platforms.

Configuring IDS

Typically, you use IDS to examine traffic at the perimeter of your network and on DMZs. However, nothing prevents you from applying IDS anywhere you need to audit suspicious network activity.

The following is a basic IDS configuration and audits traffic coming from the Internet (interface Serial0):

```
ip audit notify log
ip audit smtp spam 30
ip audit name MY-AUDIT info action alarm
ip audit name MY-AUDIT attack action alarm drop reset
!
interface Serial0
 description Interface to the Internet
 ip audit MY-AUDIT in
```

The command **ip audit notify log** tells the router to send IDS messages in syslog format. Optionally, you can send messages to a NetRanger Director central monitoring station (this is the management server for standalone NetRanger devices). Consult the IOS documentation
for NetRanger Director configuration.

The command **ip audit smtp spam 30** configures a parameter for the SMTP (e-mail) anti-spam signature. IDS monitors SMTP traffic and suspects a spam attack (junk e-mail sent to many people) if the number of recipients in a mail message exceeds this threshold (in this case, 30 recipients).

The command **ip audit name MY-AUDIT info action alarm** creates an audit rule called **MY-AUDIT** and configures IDS to send alarm messages in response to traffic matching informational signatures. Informational signatures identify suspicious, but not necessarily hostile activity.

The command **ip audit name MY-AUDIT attack action alarm drop reset** configures the action to be taken by **MY-AUDIT** when traffic matches an attack signature. The keywords **alarm drop reset** define a three-step action: Send an alarm, drop the packet, and reset the connection. Reset applies to TCP connections—IDS will send TCP packets with the reset (RST) flag set to both members of the session. Attack signatures identify suspicious and potentially hostile activity.

NOTE The IOS documentation section for IDS includes a complete list of the informational and attack signatures with descriptions.

Finally, the command **ip audit MY-AUDIT in** applies the audit rule **MY-AUDIT** to the Internet interface and specifies the inbound direction. Traffic entering Serial0 from the Internet is examined by IDS and scrutinized for suspicious activity.

TIP You can run IDS alone, CBAC alone, or IDS and CBAC together. Because IDS is an additional router task, you should evaluate your router's performance with and without it and consult Cisco for implementation assistance.

Additional Commands for IDS

Here are some additional commands for IDS configuration:

- **ip audit signature *<sig#>* disable** enables you to exclude a particular signature from the auditing process. This command disables detection of the signature identified by the signature number you specify. Cisco assigns a 4- or 5-digit number to every

signature in the signature database (consult the IOS documentation section on IDS for a list of signatures). You use this command if a particular signature alarms repeatedly (becomes an annoyance) and is not a security concern in your environment. This is called a *benign trigger.*

- **ip audit signature 1234 list 11** allows IDS to compare traffic permitted by access list **11** to signature number **1234**. Traffic denied by the access list is not compared to this signature. Use this to reduce false alarms and benign triggers. The default is to compare all traffic to each signature (unless the signature was disabled by **ip audit signature *<sig#>* disable**).
- **ip audit name MY-AUDIT info list 12** configures IDS to compare traffic permitted by access list **12** to the audit rule **MY-AUDIT**. Traffic denied by the access list is not subject to this audit rule (bypasses IDS). This keyword **list** is an optional add-on to the **ip audit name** command covered earlier. The default is to compare all traffic to the audit rule.
- Useful **show** commands: **show ip audit statistics, show ip audit configuration**, and **show ip audit interface**.
- Debugging commands: **debug ip audit ?** shows you a list of IDS debugging commands.

Summary

This chapter has extended the discussion on Cisco's advanced security services to the IOS Firewall feature set and Intrusion Detection System. Both of these services are generally deployed at the network perimeter, although nothing prevents you from using them in other parts of the network to bolster security.

The following are the key concepts of this chapter:

- The IOS Firewall feature set provides stateful application-level filtering, denial of service detection and prevention, real-time alerts, auditing of sessions, and Java applet blocking.
- The firewall service (CBAC) prevents unauthorized access to the internal network from the outside and still allows your internal users access to services on the Internet.
- CBAC allows return traffic of legitimate sessions to pass through access lists that normally prevent external traffic from entering the internal network.
- CBAC allows legitimate return traffic through to the internal network by dynamically adding temporary access list rules to existing access lists.
- The basic CBAC configuration steps: Configure and apply extended access lists to protect the internal network, define a CBAC inspection rule, apply the inspection rule to an interface.
- Security is a constant process of staying current with security advisories, implementing defenses, and auditing the effectiveness of your security systems.
- Intrusion detection provides constant surveillance, detection, termination, and alerting of suspicious network activity.