Global Information Assurance Certification Paper

Cisco Context Based Access Control (CBAC)
Jim Wiggins
January 1, 2002

Context based access control (CBAC) has been available with the Cisco Firewall Feature Set (FFS) since IOS version 11.3. Although it is only one portion of the FFS, it is one of the more valuable aspects in that it can provide an elementary form of stateful packet inspection in order to protect and preserve the internal security of a network. It is a type of extended access list, and is similar in approach to a reflexive access list. It's main advantages over reflexive access lists are that it will allow monitoring of packet traffic beyond the transport layer of the TCP/IP stack, and it has the ability to monitor traffic that may originate on one port but has continuity on another port (such as FTP). Its initial operation is akin to reflexive access lists, in that it opens temporary "holes" in defined access lists to allow return traffic, either into or out of the network. It uses a type of stateful inspection table that is stored locally (until the exchange is completed) to determine whether or not to permit network access via the router.

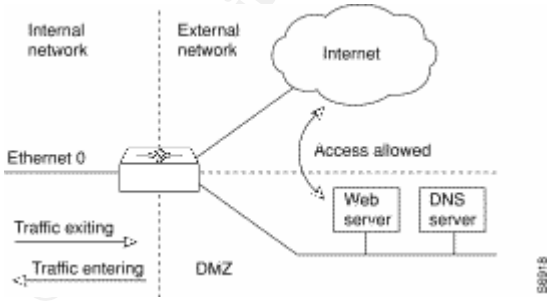
Operational Basics

As with all access lists, proper planning is the key to success. Primary considerations, such as:

a) what type of traffic to filter
b) what destinations/sources to allow/deny and
c) what interfaces will be used for the firewall

should all be calculated before any access lists are applied.

The following diagram illustrates a typical network configuration (reference 1 below):



Assuming that CBAC is enabled on the router's internal interface (Ethernet 0) in the diagram, the operation of the context based access control would be as follows:

1) Traffic is generated for an external network somewhere within the internal network attached to the Ethernet 0 port on the router.
2) After the packet reaches the router interface, it is evaluated by an access list for permission to pass. The CBAC process inspects the packet, and the source and

destination IP (Internet Protocol) addresses are recorded, along with the associated port number (this is the "state" data).

3) The data recorded is stored in a temporary state table, which exists until the session is terminated. The access list is temporarily modified by CBAC to allow return traffic matching the state table data.
4) When outside traffic destined for the internal network arrives at the router, the router checks the entry against the access list, then compares it to the state table to verify that the data is part of an eligible (previously invited) connection. The data payload of the packet is also inspected (but only on the control channel) to determine which upper layer application is being used.
5) This upper layer inspection of the payload is what allows CBAC to identify traffic based on application layer protocols, and enables CBAC to correctly allow traffic back into the network even though the outgoing port number may not be the same as the incoming port number destination (as in FTP communications). The temporary access list and state table are modified accordingly if this is the case.
6) After completion of the data transfer, the temporary modifications to the access list and the data in the state table are removed.

CBAC is very aware of the way most upper layer protocols work. It has enough intelligence to be aware of TCP sequence numbers and their probable ranges. It can also be programmed to drop half open connections to deny SYN-flood attacks. It also has the capability to judge whether UDP packets are of a similar nature, so that malformed or hacked UDP packets are dropped. Some of the application layer protocols of which CBAC processes are aware include:

CU-SeeMe (White Pine version only)	FTP
H.323 (Netmeeting, Proshare)	HTTP (Java blocking)
Java	Microsoft Netshow
UNIX r-commands (ex. rlogin, rsh)	RealAudio
RPC (Sun and Microsoft, not DCE)	SMTP
SQL *Net	StreamWorks
TFTP	VDOLive

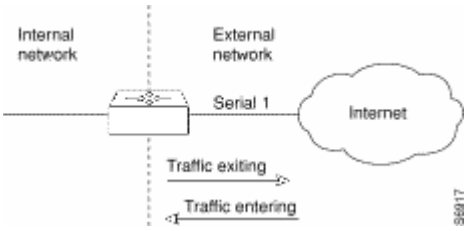
CBAC is not perfect, and has some limitations. It works only with UDP and TCP packets; other traffic must be filtered using different means such as access lists. Data that originates at the router (internal to the router) is not inspected. It can work with IPSec if the CBAC router is one endpoint of the IPSec tunnel. If encryption is being used on the data, CBAC will not be able to inspect the data payload accurately. In an ordinary network, however, the advantages of CBAC are much greater than the deficiencies.

Application of CBAC

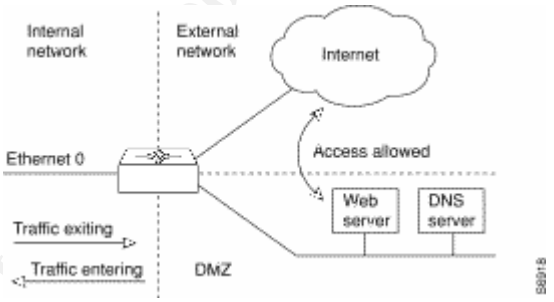
1) <u>Choose the interface</u>

The first step is to choose an interface to apply CBAC to. This can be an internal or external network interface of the firewall. Placement is determined by what you are

trying to achieve with the firewall. There are generally two scenarios. The first is to deny any traffic into an internal network, unless it is considered valid; for example, an answer to a request from the internal network. In this case, the access list and CBAC should be established on the external interface of the router. This is shown by the diagram below (Cisco, 1):



The second scenario is one in which there is a DMZ (demilitarized zone) attached to the router, to which the public has access, and a closed internal network also attached to the router. In this case, the access list and CBAC should be set on the internal interface of the router, and the router will route packets (either to the internal network or the DMZ) accordingly. The diagram below shows this (Cisco, 1):



2) Configure Access Lists

When configuring CBAC on an external interface (first scenario), the outbound IP access list can be either standard or extended. It should permit traffic that you want to be inspected by CBAC; otherwise the traffic will be dropped. The inbound IP access list must be an extended ACL (access control list), and should deny traffic that you want CBAC to inspect. Remember that CBAC will create temporary holes in this ACL as required to allow valid traffic to return to the internal network.

When configuring CBAC on an internal interface (second scenario), the inbound IP ACL at the internal interface, or the outbound IP ACL at the external interface can be either standard or extended ACLs (placement here depends on how you decide to filter traffic). They should permit traffic that you want to be inspected by CBAC, or else the traffic will be dropped. The outbound ACL at the internal interface or the inbound ACL at the external interface must be extended, and should deny traffic that you want CBAC to inspect. As stated above, it is from this end that CBAC will create temporary holes in the extended ACL to allow legitimate (previously internally generated) traffic. It is not necessary to configure extended ACLs on both the outbound internal and inbound external, but at least one extended has to be configured in order to perform CBAC.

General convention for setup of a standard access list is as follows:

Router(config)# access-list 1 permit 1.1.1.1 0.0.0.255

To apply the list to an interface:

Router(config)#interface s0
Router(config-if)#ip access-group 1 in

This ACL would allow all traffic of any type from network 1.1.1.0 into the Serial 0 interface on the router.

General convention for simple setup of an extended access list is:

Router(config)#access-list 101 permit tcp any host 1.1.1.1 eq 80 (or http)

To apply the list to an interface:

Router(config)#interface Fa0
Router(config-if)#ip access-group 101 in

This ACL would allow only HTTP traffic from specific host 1.1.1.1 into the fast Ethernet interface 0 on the router.

Note that there is always an implied deny any included at the end of all ACLs. There are many permutations to ACL usage, and it is assumed that the reader is familiar with basic ACL setup and usage; so further explanation of the details of basic standard and extended ACLs won't be covered here.

3) **Configure timeouts and thresholds**. Following is a chart taken directly from the Cisco website (Cisco, 1) that explains the commands and defines the default timers:

Timeout or Threshold Value to Change	Command	Default
The length of time the software waits for a TCP session to reach the established state before dropping the session.	**ip inspect tcp synwait-time** *seconds*	30 seconds
The length of time a TCP session will still be managed after the firewall detects a FIN-exchange.	**ip inspect tcp finwait-time***seconds*	5 seconds
The length of time a TCP session will still be managed after no activity (the TCP idle timeout).	**ip inspect tcp idle-time** *seconds*	3600 seconds (1 hour)
The length of time a UDP session will still be managed after no activity (the UDP idle timeout).	**ip inspect udp idle-time** *seconds*	30 seconds
The length of time a DNS name lookup session will still be managed after no activity.	**ip inspect dns-timeout** *seconds*	5 seconds
The number of existing half-open sessions that will cause the software to start deleting half-open sessions.	**ip inspect max-incomplete high***number*	500 existing half-open sessions
The number of existing half-open sessions that will cause the software to stop deleting half-open sessions.	**ip inspect max-incomplete low***number*	400 existing half-open sessions
The rate of new unestablished sessions that will cause the software to start deleting half-open sessions.	**ip inspect one-minute high** *number*	500 half-open sessions per minute
The rate of new unestablished sessions that will cause the software to stop deleting half-open sessions.	**ip inspect one-minute low** *number*	400 half-open sessions per minute
The number of existing half-open TCP sessions with the same destination host address that will cause the software to start dropping half-open sessions to the same destination host address.	**ip inspect tcp max-incomplete host***number* **block-time** *minutes*	50 existing half-open TCP sessions; 0 minutes

All of the commands listed above are run from the global configuration mode on the Cisco router. These commands set up the various parameters that affect the global operation of CBAC.

4) **Define the inspection rules**.

Generically, setting up CBAC is very simple. The following format is valid for all CBAC except Java and RPC (remote procedure call):

Router(config)#ip inspect name *inspection-name protocol* {alert [on|off]} {audit-trail [on|off]} {timeout *seconds*}

The italics represent user input variables.

Note that there are only 3 options: alert, audit-trail, and timeout. The alert switch allows for CBAC to send violation messages to either the router's buffer memory or to a syslog server. The audit-trail switch lets CBAC track very specific information (source and destination IP, port numbers, and number of bytes transferred) about connections used by watched applications. Timeout varies the amount of time for that particular protocol. An example setting one inspection rule is:

Router(config)#ip inspect name extfirewall ftp alert on

This creates a CBAC process called extfirewall that monitors ftp traffic, and sends alerts to buffer memory when a violation occurs.

Using CBAC for RPC, the format is as follows:

Router(config)#ip inspect name *inspection-name* rpc program-number *number* {wait-time *minutes*} {alert [on|off]} {audit-trail [on|off]} {timeout *seconds*}

Example:
Router(config)#ip inspect name extfirewall rpc program-number 11000

To configure CBAC for Java blocking, there are 2 steps involved:

Create a standard ACL that allows permitted sites.
Create a CBAC that references the ACL. The ACL is a standard list, and the format for the command for the Java blocking using CBAC is as follows:

Router(config)#ip inspect name *inspection-name* http {java-list *ACL#*} {alert [on|off]} {audit-trail [on|off]} {timeout *seconds*}

Following is an example of using CBAC for Java blocking:

Router(config)#access-list 1 permit 172.1.2.0 0.0.0.255
Router(config)#ip inspect name extfirewall http java-list 1

This allows Java applets from network 172.1.2.0, but denies them from all other networks. If there were no defined ACL, all Java applets would be denied.

If both specific and general inspection rules are applied, the specific takes precedence over the generic. In other words, if FTP and TCP were enabled for inspection by CBAC, the FTP rules would take precedence over the generic TCP rules.

5) Apply the inspection to an interface

The command to apply an inspection rule is very simple.

Router(config)#interface s0
Router(config-if)#ip inspect *inspection-name* {in|out}

Example:

Router(config)#interface s0
Router(config-if)#ip inspect extfirewall in

The only option here is whether we monitor traffic going into or out of the network. This is easily determined using the simple logic of traffic flow.

Verifying CBAC operation

There are several commands in the Cisco IOS that allow you to verify CBAC operation. All of them will not be covered here, but following are several of the most important. These commands are issued from the privileged exec mode of the router.

Show ip inspect config	Shows all specific portions of a CBAC configuration. Gives alert, timeout, and audit-trail information. Tells the name of the inspection, and rules applied.
Show ip inspect interfaces	Shows CBAC rules applied to specific interfaces, and ACLs used.
Show ip inspect session	Shows information with regard to the CBAC state table, best used with detail addition for more information.
Show ip inspect name *Inspection-name*	Shows CBAC inspected protocols.
Show ip inspect all	Shows output from all of the above commands simultaneously.

Other useful commands:

No ip inspect	Turns off CBAC
Ip inspect audit-trail	Outputs all CBAC monitored traffic to either the router buffer or to a syslog server. Not to be used lightly, it consumes a large amount of router resources and can shut down a router if improperly configured.
Logging buffered	Allows logging to the router memory buffer.
Logging *ip address*	Sends router log to designated IP address (syslog server).

Known problems with CBAC

There have been at least two warnings issued by Cisco about the usage of CBAC on Cisco routers:

1) The first had to do with IOS FFS versions 11.2 through 12.0(2)T. The problem directly related to the way that the IOS dealt with fragmented packets. There was apparently no filtering of non-initial IP fragments, and the router could be compromised through a fragmented IP packet attack. This was corrected in IOS FFS versions 12.0(2)T by keeping track of interfragment state. Non-initial IP fragments were discarded in later IOS versions.

2) The second is a much more serious bug, and was recently determined. This bug affects all versions from 11.2P to 12.1. Information about the bug was released by Cisco, and pertained to the inspection of packets entering the network as eligible traffic (a return to a request, for example). The returning packet is inspected by CBAC to see if it matches the state table information (source and destination IP addresses, and the port number (if applicable)). It does not check the return protocol type. This bug could allow traffic, with properly spoofed IP addresses and port numbers, of undetermined protocol to enter a network. This could result in very serious router hacking/takeover. As of the writing of this paper, no workaround has been determined. Cisco recommends upgrading to the latest version of their IOS (which they are allowing for free, due to the severity and nature of the problem), and they are working rapidly to fix the problem.

Summary

Context based access control (CBAC) is a very useful subset of the Cisco Firewall Feature set. Although there are other additions to the FFS, CBAC can be extremely useful in configuring a rudimentary stateful firewall inspection mechanism on a Cisco

router, which will at the very least deny a great many attempts at breaching an internal network. There are some limitations using this tool, but the advantages of a properly operating configuration far exceed the limitations. A robust active network defense would do well to use this feature of Cisco routers, and the extra cost involved for the FFS is minimal when compared to the trouble it could save a network security professional.

References

1) Cisco Systems
http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/secur_c/scprt3/sccbac.htm

2) Network Security Library Online, February 1999
http://secinf.net/info/fw/cisco/cisco.html

3) National Security Agency, Guide to Cisco Router Setup, December 27, 2001
http://nsa1.www.conxion.com/

4) Held & Hundley, Cisco Access Lists Field Guide
McGraw Hill Publishing, 2000

5) Morrissey, Peter, The cost of Security on Cisco Routers, February 1, 1999
http://www.networkcomputing.com/1004/1004ws2.html

6) Strebe & Perkins, Firewalls 24 Seven, Alameda, CA.
Sybex Network Press, 2000

7) Buchmann & Hogrell, CCNP Routing Guide, Berkeley CA.
Osborne Publishing, 2000

8) Bugtraq List
http://lists.nas.nasa.gov/archives/ext/bugtraq/1998/09/msg00099.html

9) Router God Website
http://routergod.com/donking/