Using a Router as a Firewall

It happens, one day your boss tells you one of the company's remote sites will be obtaining their own Internet access and commissions you to provide such access. The site has already installed a 2501, used to obtain access back to the corporate network. Traditionally, Internet access would require the purchase of a second perimeter router and a firewall to secure the internal network. But you already have a perfectly good router with one free serial interface. It would be nice to forgo the additional router/firewall combination and use the 2501 to connect the site to the Internet.

Or you need to secure a particular section of your internal network. Maybe you do not want your employees to have access to the Human Resources servers – the servers that contain private employee data such as salary and health care information. Does the sales department really need network connectivity to the software department's development servers? Or does your software department need access to your marketing information? By one account, 70% of network hacking attempts originate from inside the network – the part of the network not protected by your Internet firewall. While you could purchase additional stand-alone firewalls to protect your internal resources, this becomes expensive and cumbersome to maintain. In many cases the preferred solution would be to secure your network with the already installed router infrastructure.

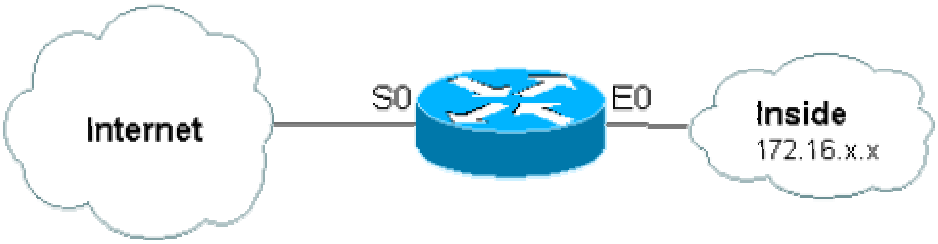
While using a router as a firewall has always been possible, the question remains can it be done securely? Traditionally the answer has been no. Standard and extended access-lists on IOS code did not maintain state information - thus limiting the effectiveness of the router as a firewall. The router had no way of knowing if a packet received at the router was a response to an internal request, or an attack disguised to look like a response to an established session. But what about the ESTABLISHED keyword at the end of an access-list? Would this not check to see if the session has been established? Not really. The ESTABLISHED keyword checks the TCP packet to determine if it is a response to an established session (i.e. is the ACK or RST bit set?). A hacker can spoof the packet to match the ESTABLISHED command, thus bypassing your access-list. A true firewall will prevent spoofing by keeping additional state information independent of the packet bit settings.

Reflexive access-lists (introduced in IOS version 11.3) went a long way in fixing the problem by keeping track of individual sessions and their states. But Reflexive access-lists were limited to the protocols that maintain a single port such as Telnet and HTTP. Protocols that required the changing of ports still did not work. An example is Active FTP (Passive FTP does not have this problem). While the FTP session is established on port 21, a second data channel is connected using port 20. A packet layer firewall does not understand this behavior and prevents the establishment of the data channel. To run such a

protocol, advanced application intelligence must be used to evaluate the traffic. Thus firewalls were still preferred over routers using reflexive access-list.

But there is another technology we may use, Context Based Access Controls (CBAC). CBAC offers true application layer intelligence in a router. With CBAC you may create detailed access-lists that filter based upon application layer traffic, so protocols such as Active FTP may be used. As of 12.2, CBAC may filter upon the following application layer protocols: CU SeeMe, FTP, H323, Microsoft's NetShow, Unix R commands, Real Audio, Remote Procedure Calls (RPC) , SQL* Net, StreamWorks, TFTP, Real Time Streaming Protocol (RTSP), and VDO Live. Plus CBAC offers advanced JAVA and SMTP filtering. With CBAC you may obtain similar firewall functionality in a router!

Let's look at a CBAC configuration using the following setup:



We will pick the RFC 1918 address of 172.16.0.0/16 for use as the inside and have been assigned the address of 206.195.128.2 from our ISP. So first we should setup network address translation (NAT) on the router to convert the invalid 172.16.x.x addresses to the valid 206.195.128.2 address.

Setup the router to translate invalid addresses to the valid address on serial 0 of 206.195.128.2 using the command:

```
ip nat inside source list 1 interface Serial0 overload
```

By using the command `overload`, you are enabling Port Address Translation (PAT) allowing all connections on the inside to use a single IP address on the outside.

The command `list 1` refers to access-list 1 which is describing what traffic should be translated:

```
access-list 1 permit 172.16.0.0 0.0.255.255
```

Apply NAT to the interfaces:

```
interface s0
   ip address 206.195.128.2 255.255.255.252
   ip nat outside

interface e0
   ip address 172.16.1.1 255.255.255.0
   ip nat inside
```

You have now allowed the inside computers to access the Internet via the address 206.195.128.2.

Because we will not be running a routing protocol from our router to the Service Provider's router, we will setup a default gateway forwarding all unknown traffic to the Service Provider:

```
ip classless
ip route 0.0.0.0 0.0.0.0 206.195.128.1
```

We now have basic Internet access. But with this configuration your network is not secure as traffic is allowed to flow in both directions, unabated.

To prevent this, let us apply an extended inbound access-list on S0. To create the list, you must realize that anything that is not permitted on the list will be denied. Therefore, you must explicitly allow your routing protocols, or any IP SEC traffic entering the router from the serial 0 interface. So, for example, if you were using BGP you must allow TCP port 179 from the BGP Neighbor to your router. If you are using IPSEC with ISAKMP you must allow protocols 50, 51 and UDP port 500. Because we are using a simple configuration with a static default gateway, you may simply say:

```
access-list 100 deny   ip any any log
```

The log command allows logging of packets that match the access-list line to your syslog server. In this case, you will see a log message for denied packets. Here is an example if a denied ICMP echo:

```
00:16:29: %SEC-6-IPACCESSLOGDP: list 100 denied icmp 109.87.60.10
-> 206.195.128.2 (0/0), 4 packets
```

These log messages will help in identifying network attacks and intrusion attempts.

Apply the access-list to the interface:

```
interface Serial0
 ip access-group 100 in
```

At this point, your phone will start ringing with co-workers who, just moments ago, were happily surfing the net and are now informing you that the "Internet is down". Do not worry through the magic of CBAC their Internet access will be restored. In the interim, ask the callers to write down exactly what they were doing before *they* broke "the Internet".

To enable CBAC you must first tell the router which protocols you desire. Most CBAC configurations allow generic TCP and UDP applications (i.e. Telnet, HTTP, NEWS etc.)

```
ip inspect name CiscoWorld tcp alert on audit-trail on
ip inspect name CiscoWorld udp alert on audit-trail on
```

The above commands created two inspection rules called CiscoWorld, one for generic TCP and a second for UDP. Notice the command "alert on", enabling alerts for this protocol. We have also enabled audit-trails to keep track as to how the connection is being used. Be careful, audit-trails can generate substantial log files.

To enable audit-trails for all protocols, use the command:

```
ip inspect audit-trail
```

Finally we will add application layer protocols:

```
ip inspect name CiscoWorld ftp audit-trail on
ip inspect name CiscoWorld realaudio audit-trail on
```

Here is a list of application layer proxies we may choose:

```
cuseeme      CUSeeMe Protocol
fragment     IP fragment inspection
ftp          File Transfer Protocol
h323         H.323 Protocol (e.g, MS NetMeeting)
http         HTTP Protocol
netshow      Microsoft NetShow Protocol
rcmd         R commands (r-exec, r-login, r-sh)
realaudio    Real Audio Protocol
rpc          Remote Prodedure Call Protocol
rtsp         Real Time Streaming 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
```

Quite an impressive list of application layer proxies available! Notice there are two that do not seem to fit, HTTP and SMTP. The HTTP proxy is used to filter

JAVA applet from HTML code. The SMTP proxy is used to filter illegal commands from the SMTP data stream, thus preventing a variety of attacks to your mail server.

Now apply the inspection rules to an interface. The inspection rules must be applied in such a way that outbound traffic is inspected. You may do this, in our configuration, by inspecting the inbound Ethernet 0 traffic or the outbound Serial 0 traffic.

```
interface Serial0
 ip inspect CiscoWorld out
```

or

```
interface Ethernet0
  ip inspect CiscoWorld in
```

If inspection is placed on the outbound serial interface, all traffic leaving Serial 0 will be inspected. Inspection on the Ethernet interface will inspect all traffic entering the Ethernet interface. While the differences are subtle, they are important. If you place the inspection rule on the inbound Ethernet interface, all traffic entering the router from a different interface (i.e. Serial 1 on a 2501) will not be inspected, and thus returning traffic will be denied. Traffic entering from the Serial 1 interface will be denied access to Internet services. Inspection on the outbound Serial 0 interface will inspect all traffic regardless of origin.

Through the inspection process, a session originating from the inside of your network will cause the inbound access-list on Serial 0 to be changed dynamically allowing the session's return traffic. You may see this by using the command `show access-list 100` while a session is in progress. Here is an example of an Active FTP session:

```
      permit tcp host 63.104.50.75 eq ftp-data host 206.195.128.2 eq
1192
      permit tcp host 63.104.50.75 eq ftp host 206.195.128.2 eq 1190
      deny ip any any log
```

Notice the two permit statements allowing the server 63.104.50.75 to connect via FTP (port 21) and FTP-data (port 20). Once the FTP session has been terminated, those two lines will be deleted.

Because we have audit-trails enabled for FTP, we will be informed of all FTP transfers as seen from the following example SYSLOG message:

```
4d09h: %FW-6-SESS_AUDIT_TRAIL: ftp-data session initiator
(63.104.50.75:20) sent 29475 bytes -- responder (172.16.1.98:1206) sent
0 bytes
```

The SYSLOG message shows the internal machine (172.16.1.98) downloaded 29,475 bytes from the server 63.104.50.75.

We now have a basic firewall established using the router. Let us add to the configuration an internal SMTP server we may use to send and receive email. While it is best to keep public servers on a DMZ, I will demonstrate the server inside the corporate network for simplicity.

First we must obtain a second valid IP address, as the SMTP server address needs to be deterministic. Assume the address of 206.195.128.5 is available. We will define the static NAT of the valid address of 206.195.128.5 to the inside address of 172.16.1.30 using the command:

```
ip nat inside source static 172.16.1.30 206.195.128.5
```

All packets with the destination address of 206.195.128.5 will be forwarded to 172.16.1.30.

We must now modify the inbound Serial 0 access-list to allow access to the SMTP server from any server in the world:

```
access-list 100 permit tcp any host 206.195.128.5 eq smtp
```

This statement will allow any address to connect to our mail server using the SMTP port (port 25). Notice you specify the destination address of the valid external address of 206.195.128.5, not the internal address of 172.16.1.30.

But now a hacker may connect to your mail server and send any command, valid or invalid, to the server in an attempt to find a security flaw. To protect the server, we should enable IP inspection for inbound SMTP traffic using the following configuration:

```
interface Serial0
  ip inspect MailServer in
!
ip inspect name MailServer smtp alert on audit-trail on
```

Now, only valid SMTP commands will be allowed through the router.

The SMTP server will also be protected against the class of denial of service attacks that involve flooding the server with half-open or embryonic TCP connections. Typing `show ip inspect config` produces the following:

```
Session audit trail is enabled
Session alert is enabled
one-minute (sampling period) thresholds are [400:500] connections
max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0
minute.
```

```
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
 Inspection name CiscoWorld
    tcp alert is on audit-trail is on timeout 3600
    ftp alert is on audit-trail is on timeout 3600
    smtp alert is on audit-trail is on timeout 3600
    realaudio alert is on audit-trail is on timeout 3600
    udp alert is on audit-trail is on timeout 30
 Inspection name MailGuard
    smtp alert is on audit-trail is on timeout 3600
```

Notice the lines:

```
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
```

The default configuration specifies if 500 embryonic connections are received, the router will start deleting connections until the number of unestablished connections falls below 400. You may change these values by using the commands `ip inspect max-incomplete high` and `ip inspect max-incomplete low`. To specify one-minute rate thresholds (i.e. number of embryonic connections in a one minute time period) use the commands `ip inspect one-minute high` and `ip inspect one-minute low`.

You may also specify the number of embryonic connections per server, and not just a total for all servers protected by the router, with the command `ip inspect tcp max-incomplete host`. The default values of `ip inspect tcp max-incomplete host 50 block-time 0` will delete the oldest embryonic connection to the host for each new connection that exceeds 50 embryonic connections. This configuration maintains 50 or less embryonic connections for each individual host.

By using CBAC to protect our mail-server, we are protected against invalid SMTP commands and denial of service attacks involving half-open TCP connections.

The complete configuration is as follows:

```
version 12.1
!
ip inspect audit-trail
ip inspect name CiscoWorld tcp audit-trail on
ip inspect name CiscoWorld ftp audit-trail on
ip inspect name CiscoWorld realaudio audit-trail on
ip inspect name CiscoWorld udp alert on audit-trail on
ip inspect name MailServer smtp alert on audit-trail on
```

```
!
interface Ethernet0
 ip address 172.16.1.1 255.255.255.0
 ip nat inside
!
interface Serial0
 ip address 206.195.128.2 255.255.255.252
 ip access-group 100 in
 ip nat outside
 ip inspect MailServer in
 ip inspect CiscoWorld out
!
ip nat inside source list 1 interface Serial0 overload
ip nat inside source static 172.16.1.30 206.195.128.5
ip classless
ip route 0.0.0.0 0.0.0.0 206.195.128.1
!
access-list 1 permit 172.16.0.0 0.0.255.255
access-list 100 permit tcp any host 206.195.128.5 eq smtp
access-list 100 deny   ip any any log
!
end
```

This router/firewall may also perform advanced firewalling capabilities such as the establishment of Virtual Private Networks (VPN) using IP Sec, and authentication with a AAA server. The Cisco Secure Policy Manager software and InfoSim's (www.infosim.com) StableNet ARC product can assist with the creation and management of VPNs on the router. Some routers even have the capability of rudimentary Intrusion Detection thus distributing your intrusion detection to the router level.

And there you have it, a fully functional firewall created using the IOS CBAC commands. The firewall utilizes state tables for added security and has application layer proxies for many common protocols. Plus the firewall creates detailed logs of failed connections, suspicious activity, and successfully completed sessions.

So should you replace your firewalls with routers? Not necessarily. I prefer to use a firewall, such as the PIX, in situations that traditionally require a firewall (i.e. your Internet access point). I prefer to use the IOS firewall features at locations inside my network that already have routers and require additional security.

My reasoning is due to the basic design of a firewall versus a router. On the PIX firewall, the default behavior is to allow all inside traffic out and deny all outside traffic in (unless the outside traffic is a response to a connection made from the inside). This configuration may be accomplished with a minimum number of statements. If you would like to modify this behavior, you must add additional conduits or access-lists. A router with a minimal configuration will allow all traffic

and must be secured with additional statements. While they may both be made secure, I argue that an incorrect configuration on a router is more likely to cause a security breach then on a PIX. Where the IOS firewall features shines is securing and segmenting the internal router based network, or protecting a small office or home office network. My network designs usually involve using a PIX firewall to secure the corporate Internet access and public servers on a DMZ. The routers IOS security features are used to segment the internal network thus creating additional barriers against attack.

About the Author

Paul Borghese (CCIE # 3760) is a network consultant with GTSI and a contract instructor for Global Knowledge Networks. Mr. Borghese has over 12 years experience in the networking field specializing in network security systems and design. He is the founder and owner of GroupStudy.com, the Internet's largest and most active Cisco Certification discussion group. Mr. Borghese is currently the author and course director for Global Knowledge's CCIE Security preparation curriculum.