Global Information Assurance Certification Paper

Richard Meehan
SANS Security DC 2000

Firewall and Perimeter Protection
"Network Defense for the Top Ten Vulnerabilities"

1. Introduction

A network perimeter protection strategy is an implementation of a security policy. The perimeter defense solution discussed in this paper implements a security policy that was designed based on the top ten vulnerabilities released by SANS and posted at <http://www.sans.org/topten.htm>. Perimeter protection is provided by named access control lists (ACLs) on a Cisco 3640 router running IOS 11.2.

The security policy requires ports that are commonly probed and attacked to be blocked. Realize that blocking these ports is a minimum requirement for perimeter security, not a comprehensive firewall specification list. A far better rule is to block all unused ports and actively monitor them to detect intrusion attempts; however, this is beyond the scope of this paper. This design is limited to only blocking the ports in the security policy as this can be effectively accomplished with ACLs. Blocking some of the ports required by the policy may disable needed services. The potential effects of blocking, and hence, disabling these services will be discussed as appropriate.

2. The Security Policy

The security policy requires the following actions:

1. Block "spoofed" addresses-- packets coming from outside the local area network that have source addresses from internal addresses or private (RFC1918 and network 127) addresses. Also block source routed packets.
2. Block login services-- telnet (23/tcp), SSH (22/tcp), FTP (21/tcp), NetBIOS (139/tcp), rlogin et al (512/tcp through 514/tcp).
3. Block RPC and NFS-- Portmap/rpcbind (111/tcp and 111/udp), NFS (2049/tcp and 2049/udp), lockd (4045/tcp and 4045/udp)
4. Block NetBIOS in Windows NT -- 135 (tcp and udp), 137 (udp), 138 (udp), 139 (tcp). Windows 2000 - earlier ports plus 445(tcp and udp)
5. Block X Windows -- 6000/tcp through 6255/tcp
6. Block naming services-- DNS (53/udp) to all machines which are not DNS servers, DNS zone transfers (53/tcp) except from external secondaries, LDAP (389/tcp and 389/udp)
7. Block mail-- SMTP (25/tcp) to all machines, which are not external mail relays, POP (109/tcp and 110/tcp), IMAP (143/tcp)
8. Block Web-- HTTP (80/tcp) and SSL (443/tcp) except to external Web servers. Also consider blocking common high-order HTTP port choices (8000/tcp, 8080/tcp, 8888/tcp, etc.)
9. Block "Small Services"-- ports below 20/tcp and 20/udp, time (37/tcp and 37/udp)

10. Block the following miscellaneous ports-- TFTP (69/udp), finger (79/tcp), NNTP (119/tcp), NTP (123/tcp), LPD (515/tcp), syslog (514/udp), SNMP (161/tcp and 161/udp, 162/tcp and 162/udp), BGP (179/tcp), SOCKS (1080/tcp)
11. Block the following types of ICMP traffic-- incoming echo request (ping and Windows traceroute), outgoing echo replies, time exceeded, and unreachable messages

3. Discussion of the Security Policy

The discussion in section 3 of the 11 filters outlined in the security policy provides details for the following areas:

1. The reason the services might be considered a vulnerability.
2. Relevant information about the behavior of the protocol or service on the network
3. The syntax of the filter as implemented on a Cisco router.
4. A description of each of the parts of the filter.

Throughout each section additional tips are provided when useful. Section 4, "Ordering the Rules", discusses the recommended order to apply all of the rules and the reasons for that order. Section 5, "Applying and Testing the Filter", discusses how the complete filter list should be applied to the router and how to test the individual filters within the entire list.

3.1 "Spoofed" addresses and Source Routed Packets

A packet that has a source Internet Protocol (IP) address that has been altered by a user so it is different from the actual IP address of the machine where the packet originated is said to have a spoofed address. Spoofing the source address of packets permits attackers to perform reconnaissance and launch attacks while not revealing the actual address of the machine they are using. Some attackers will spoof a source address by "borrowing" a valid address from a large company such as Microsoft or Sun Microsystems; thus the source address appears legitimate.

Replies to packets with spoofed addresses don't return to the location the packet originated from, but instead are sent to the spoofed address. If an attacker crafts a packet that appears to have originated from a Microsoft IP address, a reply to that packet will be sent to the Microsoft machine. There are certain addresses that should never originate external to a network because replies wouldn't reach any host; these are definitely spoofed packets and should be dropped. The most obvious address space that should not be sending packets from an external network to the internal network is the address space of the internal network; this is definitely invalid traffic. Another address that should never be an external source address is any address on the 127.0.0.0 network since this network is reserved for loopbacks.

Other addresses that should never originate outside the internal network are those addresses defined in Request for Comments (RFC) 1918, "Address Allocation for Private Internets" available at <http://www.rfc-editor.org/rfc/rfc1918.txt>. The address spaces defined in RFC 1918 should never appear on the Internet. If a client machine is using an address mentioned in

the RFC and that client needs access to services outside their local networks, there must be a gateway or proxy used which will provide a globally unique IP address. The addresses reserved for private use by RFC 1918 include the following:
10.0.0.0 - 10.255.255.255
172.16.0.0 - 172.31.255.255
192.168.0.0 - 192.168.255.255

Figure 3.1-1 shows an example of three packet source addresses originating from the most common external network, the Internet. These three source addresses illustrate the three different types of addresses that are always spoofed packets if they appear on the external side of a network. These packets should never be permitted to enter the internal network since they can never be legitimate traffic.

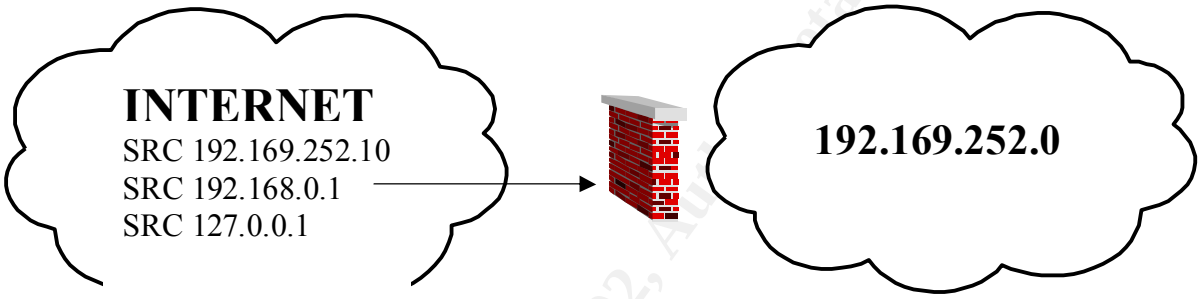

Figure 3.1-1

Packet headers usually contain two addresses, a source and a destination. The packet relies on routers between the source and destination to determine the network path the packet will follow. Source routed packets contain additional information which specifies the route that the packet should follow from the source to the destination instead of using the router's routing table. A source-routed packet usually originates from an attacker who has found an alternate network path that circumvents security measures that are in place at the expected entrance to the network. Dropping source-routed packets at the network perimeter makes it more difficult for an attacker to bypass the intended network entrance if an alternate path happens to be less protected.

3.1.1 Network Architecture

The network that will be used to illustrate the filters in this paper is a simple network that has a Cisco 3640 router with three Ethernet interfaces in use. One interface connects to a wide area network (for simplicity this will be called the Internet, although it could be a large private wide area network), one interface connects to the internal network, and one interface connects to a perimeter network. The complete network is shown in figure 3.1.1-1. Note that although the entire 192.169.252.0 address space is owned, the network on e0/1 only uses the subnet 192.169.252.8/29.

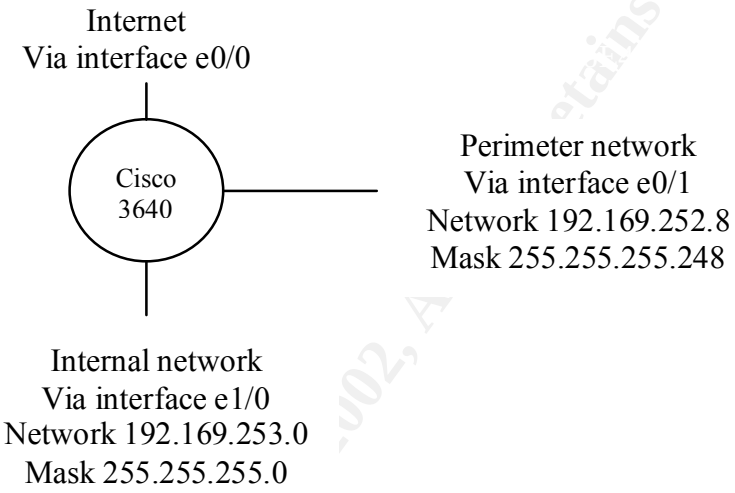

Figure 3.1.1-1 Network used to illustrate port filtering rules.

3.1.2 Filter Syntax and Description

Each of the filter rules that will be described throughout the subsections of section 3 will be used to build a named extended access control list in section 5 "Applying and Testing the Filter". The rules all follow the same syntax. This syntax is discussed below. Although the example below is wrapped across several lines, each filter is entered continuously on it's own line when configuring the Cisco router. The return key is only pressed after a line is fully entered. It is not efficient or recommended that lines be entered individually when configuring the filter. The entire filter can be written in a text editor and entered into the router via a simple cut and paste. This will limit errors caused by mistyping keys since there is no way to modify individual lines in a Cisco access control list after the line is entered. Since a named extended access list is used, the familiar notation of "access-list 101" is not needed at the start of each line.

```
[permit | deny] [protocol] [source address] (wildcard mask)
(operator and port) [destination address] (wildcard mask)
(operator and port)
```

The [permit | deny] portion of the statement is the action that will be taken when a packet matches the filter specified on that line.

The [protocol] is the protocol that the rule applies to (such as tcp, udp, icmp, ip, etc.).

The [source address] is the IP address that the packet originated from. This can be a specific IP address, a block of IP addresses, or even the word "any" to specify that all IP addresses match.

The (wildcard mask) is an optional entry (as are all the entries in the syntax that are shown in parentheses). It is used to specify what part of the source packet IP address must match the IP address specified in the filter. The significance of the bits is the exact opposite of a subnet mask. A 0 is an interesting bit (the value of the source packet IP address must match the one specified in the filter) and a 1 is an uninteresting bit (the value of that bit in the IP address can be 0 or 1). For example, if the source address in the filter was 192.168.10.8 and the wildcard mask was given as 0.0.0.7 then the IP addresses from 192.168.10.8 through 192.168.10.15 would all match the source address. A wildcard mask of 0.0.0.0 means an exact match; alternately, the word "host" can be placed in front of the source or destination address to specify one single IP address without using the wildcard mask.

The (operator and port) is an optional entry. It is used to specify a particular port or range of ports.

The [destination address] is the IP address that the packet is destined for. This can be a specific IP address, a block of IP addresses, or even the word "any" to specify that all IP addresses match.

The (wildcard mask) in this instance has the same function as the wildcard mask for the source address.

The (operator and port) in this instance has the same function as the operator and port for the source address.

The filters needed to implement the requirements of this section are as follows:

```
deny ip 192.169.252.0 0.0.1.255 any
deny ip 10.0.0.0 0.255.255.255 any
deny ip 172.16.0.0 0.15.255.255 any
deny ip 192.168.0.0 0.0.255.255 any
deny ip 127.0.0.0 0.255.255.255 any
```

When the five filters above are included in an access control list that is applied to interface e0/0 (inbound), they will prevent all IP traffic with a source address in the private address spaces from entering the local network. Local network is used to denote both the internal network and the perimeter network. To ensure the use of the wildcard mask is understood as previously explained I will use the first deny line as an example. The number 252 in binary is 11111100 and the number 253 is 11111101. The first 7 high order bits are the same for both 252 and 253.

If the last bit is a 0 or a 1, the third octet will still match so the last bit is uninteresting (while the first 7 in the third octet must be an exact match). Since all IP addresses that start 192.169.252 and 192.168.253 are on the local network, the entire last octet is uninteresting or all 1s (value 255). This gives the wildcard mask 0.0.1.255 which filters for both networks with one line.

At the very end of the access list a rule will be entered to permit any traffic to any destination. Although this is a bad thing, it is required to meet the policy of only blocking specific ports. Without permitting all traffic at the end of the ACL, the Cisco implicit "deny all" will filter anything not matching a previous rule. The implicit deny works much better when the security policy only permits certain ports/protocols and requires all others to be denied.

For interface E0/1 there will be different rules in a different access control list. There will be far fewer rules than in the access list for the external interface. The interaction of the rules will be much clearer in sections 4 and 5 when both access lists are shown with all rules in order. For now, accept the fact that spoofed addresses will be prevented from leaving the perimeter network because all statements will specify the perimeter subnet (or a specific perimeter host) as the source in the first part of the filter as follows:

```
permit [protocol] 192.169.252.8 0.0.0.7 [destination]
```

OR

```
permit [protocol] host 192.169.252.X [destination]
```

Since Cisco access lists have an implicit deny at the end of them, if traffic doesn't match the source address (as well as all the other criteria in the rule), it will be dropped. Since the ACL on interface e0/1 will keep the implicit deny in place, spoofing will not be possible from the perimeter subnet. Good practice would also call for an egress filter to prevent spoofed addresses from leaving the internal network; however, the only egress filtering discussed in this paper is on ICMP traffic to meet requirement 11 of the security policy.

Blocking source routed packets is not an access list entry, but is a configuration entry. In global configuration mode, issue the command `no ip source-route`. This will cause the router to ignore source route information

3.2 Login services

Login services are useful to attackers because at a minimum they can provide valuable information, but even more importantly they provide remote access to target computers. A simple telnet to a machine can reveal the version of the operating system in use as shown in figure 3.2-1.

```
SunOS 5.7

login:
```

Figure 3.2-1

Unless the system administrator has modified the banner in hopes of misleading novice hackers, the attacker need only search the Internet for exploits related to the identified operating system and focus on the version in use. Replacing the operating system banner with a custom banner such as the one in figure 3.2-2 is a good practice, but this doesn't correct the major flaw with these login services. Except for secure shell (SSH), all data, to include user authentication, is passed in the clear.

```
Unauthorized access to this computer is in
violation of Article 27, Sections 45A and 146 of
the Annotated Code of Maryland and will be
prosecuted to the full extent of the law.  All
usage of this system is monitored for security
purposes, and by signing on to the system you
are implicitly consenting to this monitoring.

login:
```

Figure 3.2-2

Passing user authentication in the clear allows attackers to sniff passwords for valid user accounts. While this may not yield immediate root access, it provides an easy method to obtain an account (or several accounts) that provides remote access. The attacker need only exploit a vulnerability on the system to escalate his privileges to root.

From this discussion it would appear that since SSH encrypts user authentication that allowing it into the network would not adversely affect security. This is not entirely true. SSH is a better solution than the unencrypted remote access methods when remote access is required; however,

if remote access is not required, it should be blocked. Allowing SSH when it is not required does two things, it leaves a hole in the perimeter and if SSH is running inside the perimeter it provides an attacker a service to exploit that provides interactive logins. Another point to stress is that opening port 22 to tcp traffic doesn't just allow SSH, it allows all traffic on port 22. If an attacker knows an organization allows port 22 into the internal network, he can use a tool such as nmap <http://www.insecure.org/nmap/index.html> to scan the internal network through port 22. This also points out a weakness of port filtering with a router: the packet filter does not validate that the contents of the packet are legitimate. Also, if a service is listening on a non-standard port, it will still be available even if the well-known port is blocked.

3.2.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny tcp any any eq 23
deny tcp any any eq 22
deny tcp any any eq 21
deny tcp any any eq 139
deny tcp any any range 512 514
```

When the filters above are included in an access control list that is applied to interface e0/0 (inbound), they will prevent all tcp traffic from entering the local network over the ports specified. These are the well-known ports for the services in requirement 2 of the security policy.

3.3 RPC and NFS

Remote procedure calls (RPC) allow programs to run on one computer and execute programs on a remote system. When an RPC service is started it registers with the portmapper (rpcbind). Clients contact rpcbind and are assigned a dynamic port for interacting with the requested listening RPC service. Common RPC services include ttdbserverd (ToolTalk), rpc.cmsd (Calendar Manager), and network information system (NIS), as well as those used for NFS, such as rpc.statd, mountd, and lockd. To make matters worse, many RPC services run with root privileges. Exploiting one of these can provide a fast avenue to root access. The book, Hacking Exposed: Network Security Secrets and Solutions (with a companion website available at <http://www.hackingexposed.com/>) by Stuart McClure, Joel Scambray, and George Kurtz, methodically outlines how easily RPC services are exploited (as well as discussing many other vulnerabilities associated with the services outlined in this paper's security policy. This is often due to misconfiguration or running an RPC service when it is not used since some RPC services are enabled on boot by default. Blocking these services from the external network provides a layer of insulation that makes it more difficult for an attacker to exploit them or to determine if they are even available for exploit.

3.3.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny tcp any any eq 111
deny udp any any eq 111
deny tcp any any eq 2049
deny udp any any eq 2049
deny tcp any any eq 4045
deny udp any any eq 4045
```

When the filters above are included in an ACL that is applied to interface e0/0 (inbound), they will prevent udp and tcp traffic from entering the local network over the ports specified. These are the well-known ports for the services in requirement 3 of the security policy.

3.4 NetBIOS in Windows NT

Allowing NetBIOS traffic outside your local network perimeter provides attackers an easy method to gain information about Windows NT targets on the network. Windows computers freely disclose such information as domain names, names of computers in the domain, names of domain controllers, share names, and user and group account names via the NetBIOS ports. All of this information is very useful as an attacker tries to locate his next target. Again, the Hacking Exposed book is a great resource for clearly pointing out the ease with which NetBIOS can be used for malicious purposes.

3.4.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny tcp any any eq 135
deny udp any any eq 135
deny udp any any eq 137
deny udp any any eq 138
deny tcp any any eq 139
deny tcp any any eq 445
deny udp any any eq 445
```

When the filters above are included in an access control list that is applied to interface e0/0 (inbound), they will prevent udp and tcp traffic from entering the local network over the ports specified. These are the well-known ports for the services in requirement 4 of the security policy.

3.5 X Windows

X is dangerous because access is either not allowed at all or allowed completely. The authentication mechanism is based only on IP address. Since maintaining lists of valid IP addresses that should be able to run X sessions can be a time consuming task, some administrators allow all machines access via `xhost +`. Some X-servers are even configured in this way by default! An attacker can use X windows to view any window on a remote system and observe what is happening without the user at the console knowing.

3.5.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny tcp any any range 6000 6255
```

When the filter above is included in an access control list that is applied to interface e0/0 (inbound), it will prevent all tcp traffic from entering the local network over the ports specified. These are the well-known ports for the service in requirement 5 of the security policy.

3.6 Naming Services

Many services and protocols (such as SMTP, telnet, and FTP) depend on DNS to function and allow users easy access to their capabilities. While the casual observer thinks DNS merely provides host name resolution, DNS could really be giving away more information than anyone in your company intended. An attacker may try to use the information in your DNS to determine local hostnames and IP addresses, and key on mail exchange records for your organization. Many DNS servers are misconfigured and allow anyone to perform a zone transfer. This provides an attacker with entirely too much information too easily. A patient attacker could perform manual lookups to determine information about your company, but that would take much more time. A DNS server can also be used in an attack if the attacker populates it with erroneous information. A good way to avoid this is to implement a split DNS. A DNS on the internal network would resolve internal hostnames and a DNS on a protected subnet (perimeter network) would provide resolution for hosts and services that must be placed in a less protected (usually Internet accessible) area. This prevents an attacker from querying your DNS for information about your internal network. The internal DNS can be configured as a forwarder and forward internal requests that require external resolution to the server on the perimeter network. The perimeter DNS can find the answer and return it to the internal DNS. Two resources with more details on this configuration are:
Building Internet Firewalls (http://www.oreilly.com/catalog/fire2/) and
DNS and BIND (http://www.oreilly.com/catalog/dns3/).
Operating with DNS in this fashion also alleviates the need to do any zone transfers through any filters.

There is no need for any system on a wide area network to send DNS information into your network to any machine except your perimeter DNS server. Only your local DNS should provide resolution to requests originating at your network.

LDAP is used to maintain directory databases and can be thought of as a utility protocol that programs use to look up information. Hackers can use LDAP to access directory information that reveals sensitive site information including user account names. LDAP is especially dangerous over a wide area network because the information is passed unencrypted which also provides an accessible target for hackers who like to sniff traffic.

3.6.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
permit udp any host 192.169.252.10 eq 53
deny udp any any eq 53
deny tcp any any eq 53
deny tcp any any eq 389
deny udp any any eq 389
```

These filters require a little more explanation since the ordering makes them slightly more complicated than the filters in the earlier sections. The first filter permits any Internet machine to query only the DNS on the perimeter network (192.169.252.10) while the next line denies all DNS requests to any other machine on the local network. Tcp/53 traffic is blocked with the next deny line. This will prevent anyone from doing a zone transfer. If DNS queries need to return data greater than 512 bytes they will need to be able to talk tcp over port 53 to the DNS server on the perimeter network. This is done by adding a
`permit tcp any host 192.169.252.10 eq 53`
line after the first permit rule above. A corresponding rule will be needed on the e0/1 interface ACL to allow the response to travel from the perimeter subnet to the internal network. The DNS on the internal network should be configured as a forwarder with the DNS on the perimeter network actually resolving hostnames. All machines on the internal network point to the internal DNS (192.169.253.10) for resolution.

For the perimeter DNS to pass DNS information into the internal DNS, the filter below must be added to the access list that is applied to interface e0/1 (inbound)

```
permit udp host 192.169.252.10 eq 53 host 192.169.253.10
```

3.7 Mail

All mail should be sent through a mail gateway that is the only machine that talks to other mail servers. Email has become a common transmission method for dangerous viruses; it is essential to have a controlled entryway for this potentially dangerous traffic. POP and IMAP are protocols for users to retrieve mail from the mail server. Authentication information and data are

sent unencrypted with these protocols. If these protocols are allowed in from outside your perimeter, users will be encouraged to check their e-mail across the wide area network. This can provide valid logon information to anyone sniffing traffic. Having a valid user account and password means an attacker need only find a way to escalate privileges on one of your servers before he owns your network. If possible for your site, set up mail just as you did DNS--using an internal server and, if e-mail outside the company is needed, an external server on your protected subnet.

3.7.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
permit tcp any host 192.169.252.11 eq 25
deny tcp any any eq 25
deny tcp any any eq 109
deny tcp any any eq 110
deny tcp any any eq 143
```

When the filters above are included in an ACL that is applied to interface e0/0 (inbound), they will prevent all tcp traffic from entering the local network over the ports specified except for traffic on port 25 to the server (192.169.253.11) in the perimeter network running the SMTP service. These are the well-known ports for the services in requirement 7 of the security policy.

All mail is delivered to/from the Internet to the 192.169.252.11 machine which in turns sends it to (and receives it from) the internal network (the SMTP gateway at 192.169.253.11). The filters that will be used in the access list for interface E0/1 will be

```
permit tcp host 192.169.252.11 eq 25 host 192.169.253.11 established
permit tcp host 192.169.252.11 host 192.169.253.11 eq 25
```

The first filter allows 192.169.252.11 to reply to a connection established by the 192.169.253.11 machine (mail being sent from the internal network to the Internet). The established word used at the end of the line means the ACK bit must be set. If "established" was not used, a connection could be initiated to any port on the internal mail server as long as it came from the mail server on the perimeter and the source port was 25. This would allow a hacker an easier way into the internal network if the perimeter mail server is compromised.

The second line allows the perimeter mail server to send mail into the internal server (mail coming from the Internet).

3.8 Web

If you must provide a web presence, and most companies will, then run the web server on the protected subnet with the second DNS and second mail server. Only allow the common web ports (80, 443, 8000, 8080, 8888) to this server. Web servers are another favorite target for hackers. The common web servers all have many exploits available for download that will allow anyone to do damage if the patches aren't up to date and the machine isn't configured as a bastion host.

3.8.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
permit tcp any host 192.169.252.12 eq 80
permit tcp any host 192.169.252.12 eq 443
deny tcp any any eq 80
deny tcp any any eq 443
deny tcp any any eq 8000
deny tcp any any eq 8080
deny tcp any any eq 8888
```

When the filters above are included in an access control list that is applied to interface e0/0 (inbound), they will permit connections from any Internet machine to the web server on the perimeter network only (192.169.252.12) all tcp traffic destined anywhere else over the listed ports is blocked. These are the well-known ports and common high ports for the services in requirement 8 of the security policy.

HTTP traffic is not permitted from the server on the perimeter network to any machines on the internal network since an internal server maintains the same information.

3.9 "Small Services"

These ports are rarely used for legitimate purposes and can be used to create annoying attacks. For instance, a spoofed packet that appears to be sent from the echo port of a computer on your network can be sent to the echo port of another computer and the two computers will talk back and forth to each other endlessly. Some of the common "small services" are echo, chargen, discard, and daytime. Go a step beyond blocking these at the perimeter and disable them on your servers. Additionally, small services are enabled by default on some Cisco routers; they should be disabled on the routers too.

To disable the services on the router used in this example simply issue the following commands in Global configuration mode:

```
no service udp-small-servers
```

```
no service tcp-small-servers
```

There are many other changes that should be made to help protect the router. Two good resources to start with are http://www.insecure.org/news/P55-10.txt and www.cisco.com/warp/public/707/21.html.

3.9.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny ip any any lt 21
deny tcp any any eq 37
deny udp any any eq 37
```

When the filters above are included in an ACL that is applied to interface e0/0 (inbound), they will prevent all ip traffic from entering the local network over the ports 20 and below (lt 21 is less than 21), as well as all tcp and udp traffic over port 37. These are the well-known ports for the services in requirement 9 of the security policy.

3.10 Miscellaneous Ports

There are many ports that can be dangerous for various reasons. These include the ports associated with TFTP (69/udp), finger (79/tcp), NNTP (119/tcp), NTP (123/tcp), LPD (515/tcp), syslog (514/udp), SNMP (161/tcp and 161/udp, 162/tcp and 162/udp), BGP (179/tcp), and SOCKS (1080/tcp). There are many other dangerous miscellaneous ports of course, but the discussion in this paper is limited to these.

TFTP is just that, trivial FTP. There is no authentication involved. It's often used for booting routers with configuration files across a network. Having this type of information available to anyone, especially without authentication, is a bad practice.

The finger service is just one more easy method for attackers to look up user information. Why make things easy for outside attackers? If you must enable finger (yes, you can read that as making things easier for the internal attacker) then at least block it at the perimeter.

Permitting network news transfer protocol into your network should only be permitted if you are running your own news server that has only public news groups. If that is a requirement, only open holes to the specific servers that you will transfer news with.

The network time protocol doesn't use any authentication by default so an attacker could forge packets and disrupt the time on your network. While blocking NTP may prevent an attacker from disrupting the time on your network, it won't prevent all your computers from having the same wrong time if you don't have an accurate time source available. Unsynchronized and inaccurate time can become a bother when analyzing logs.

LPD is another service that offers no user authentication or data protection. Leaving this port open across the WAN can leave a site susceptible to attacks ranging from crashing the print server to deleting files to running commands as root.

A syslog server is a great way to centrally manage log messages. If an attacker can reach that syslog server though he can flood it and use all available space which will cause logging to cease. This helps the attacker cover his tracks. Even if logging doesn't stop, there could be so much information in the log that the attack is lost in all the information.

Simple network management protocol can provide great information that helps an operations center manage and troubleshoot a network, but it can also be extremely dangerous. Because default community strings or easily guessed community strings are often used, at a minimum an attacker can read performance information about your network and also get a good idea of the resources available. If the attacker gains write access, he can cause devastation to the hardware on your network by changing configurations.

Border gateway protocol can be used to manipulate routing tables on your network's premise router. If you only have one connection to the Internet, there's no need to let BGP update routing tables, use a default route. If you have multiple connections though, BGP will be useful for ensuring the alternate route is used when needed. Updates should only be accepted from the routers that are directly connected to your router. A floating static route could also be used by the truly paranoid.

SOCKS allows standard TCP client programs (like telnet) to be converted to proxied versions of the same program. If an attacker installs a version of a client program converted to use SOCKS they can have connections sent through a rogue server and continue with more information gathering.

3.10.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny udp any any eq 69
deny tcp any any eq 79
deny tcp any any eq 119
deny tcp any any eq 123
deny tcp any any eq 515
deny udp any any eq 514
deny tcp any any eq 161
deny udp any any eq 161
deny tcp any any eq 162
deny udp any any eq 162
deny tcp any any eq 179
deny tcp any any eq 1080
```

When the filters above are included in an access control list that is applied to interface e0/0 (inbound), they will prevent all udp and tcp traffic from entering the local network over the ports specified. These are the well-known ports for the services in requirement 10 of the security policy.

3.11 ICMP

An attacker can use incoming echo requests (pings) to map your network and determine the IP addresses being used on your network. Echo requests can also be used to launch a denial of service attack, or even worse, the attacker may use a program such as Loki to tunnel commands through your perimeter filter if ICMP echo requests are allowed into your network. ICMP echo requests are helpful to network administrators troubleshooting connectivity to resources, but allowing them into your network when programs such as Loki exist can be very dangerous. If echo requests must be permitted, consider only permitting them from known IP addresses that will be used for specific functions (such as network troubleshooting). If echo requests aren't allowed into the network, there should be no echo replies to go out; blocking them reinforces this. Of course, if you open holes for echo requests for a network administrator to troubleshoot realize that a rule must also permit replies to the corresponding address. An unsolicited, outbound echo reply is definitely hostile traffic (either an attack or a channel the attacker is using to covertly transmit data). Since traceroutes are blocked inbound also, unreachable messages don't need to be permitted out from the internal network either. They provide one more bit of information an attacker could use to help determine your site security policy when using a tool such as Firewalk (which is available at http://www.packetfactory.net/firewalk/).

3.11.1 Filter Syntax and Description

The filters needed to implement the requirements of this section are as follows:

```
deny icmp any any echo
```

When the filter above is included in an access control list that is applied to interface e0/0 (inbound), it will prevent all icmp echo requests from entering the local network. Notice there is no operator such as eq needed for this line; Cisco uses an icmp-message instead to make configuration easier.

To block outgoing echo replies, time exceeded, and unreachable messages, the following filters must be specified in an access control list that will be applied to interface e0/0 (outbound):

```
deny icmp any any echo-reply
deny icmp any any time-exceeded
deny icmp any any unreachable
```

4. Ordering the Rules

Filtering rules will provide a false sense of security if the order of the rules is not carefully considered. This section discusses the recommended order to apply all of the rules and the reasons for that order.

For interface e0/0 (inbound) the rules should be ordered as follows (comments are preceded by a "!"; these would not appear in the actual configuration):

```
! immediately deny any spoofed packets
deny ip 192.169.252.0 0.0.1.255 any
deny ip 10.0.0.0 0.255.255.255 any
deny ip 172.16.0.0 0.15.255.255 any
deny ip 192.168.0.0 0.0.255.255 any
deny ip 127.0.0.0 0.255.255.255 any
! begin deny rules to prohibit traffic not allowed by the security policy
deny tcp any any eq 23
deny tcp any any eq 22
deny tcp any any eq 21
deny tcp any any eq 139
deny tcp any any range 512 514
deny tcp any any eq 111
deny udp any any eq 111
deny tcp any any eq 2049
deny udp any any eq 2049
deny tcp any any eq 4045
deny udp any any eq 4045
deny tcp any any eq 135
deny udp any any eq 135
deny udp any any eq 137
deny udp any any eq 138
deny tcp any any eq 139
deny tcp any any eq 445
deny udp any any eq 445
deny tcp any any range 6000 6255
! must permit allowable DNS traffic before denying the rest of the DNS traffic
permit udp any host 192.169.252.10 eq 53
deny udp any any eq 53
deny tcp any any eq 53
deny tcp any any eq 389
deny udp any any eq 389
! must permit allowable mail traffic before denying the rest of the mail traffic
permit tcp any host 192.169.252.11 eq 25
deny tcp any any eq 25
deny tcp any any eq 109
deny tcp any any eq 110
```

```
deny tcp any any eq 143
```
! must permit allowable web traffic before denying the rest of the web traffic
```
permit tcp any host 192.169.252.12 eq 80
permit tcp any host 192.169.252.12 eq 443
deny tcp any any eq 80
deny tcp any any eq 443
deny tcp any any eq 8000
deny tcp any any eq 8080
deny tcp any any eq 8888
deny ip any any lt 21
deny tcp any any eq 37
deny udp any any eq 37
deny udp any any eq 69
deny tcp any any eq 79
deny tcp any any eq 119
deny tcp any any eq 123
deny tcp any any eq 515
deny udp any any eq 514
deny tcp any any eq 161
deny udp any any eq 161
deny tcp any any eq 162
deny udp any any eq 162
deny tcp any any eq 179
deny tcp any any eq 1080
deny icmp any any echo
```
! if the security policy doesn't deny the traffic it should be allowed
! the next line lets everything else (not denied above) go anywhere.
```
permit ip any any
```

For interface e0/0 (outbound) the rules should be ordered as follows (comments are preceded by a "!"; these would not appear in the actual configuration):

! deny the icmp traffic that should not leave the network
```
deny icmp any any echo-reply
deny icmp any any time-exceeded
deny icmp any any unreachable
```
! allow any other traffic to leave
! it's good policy to replace the first any in the statement below with any
! internal network addresses to prevent spoofing.
! in this case the line would be "permit ip 192.169.252.0 0.0.1.255 any"
! since this was not required by the security policy no change was made
```
permit ip any any
```

For interface e0/1 (inbound) the rules should be ordered as follows (comments are preceded by a "!"; these would not appear in the actual configuration):

! permit specific host traffic to the internal network
```
permit udp host 192.169.252.10 eq 53 host 192.169.253.10
permit tcp host 192.169.252.11 eq 25 host 192.169.253.11 established
permit tcp host 192.169.252.11 host 192.169.253.11 eq 25
```
! deny all other traffic to the internal network
```
deny ip any 192.169.253.0 0.0.0.255
```
! permit any perimeter traffic to the external network
```
permit ip 192.169.252.8 0.0.0.7 any
```
! That covers all legitimate traffic so deny everything else.
! This filter that will be used on e0/0 shows how permitting only some traffic
! and denying the rest can shorten the ACL considerably.
! This is also a much better practice.
```
deny ip any any
```

5. Applying and Testing the Filter

Once the rules that will be used to make up the ACL have been determined, the ACL can be built. We'll be using an extended named ACL instead of an extended numbered ACL. One of the nice features of the named list is you can delete a line anywhere in the list by using the word "no" in front of the line (just like Cisco commands) and you can add lines at the end of the list. You will still need to enter the list over again to add lines to anywhere except the bottom. You can get creative and delete the last permit line, add a new deny line, then add the permit line back. If you are not familiar with named access lists review http://www.cisco.com/univercd/cc/td/doc/product/software/ios112/112cg_cr/5cbook/5cip.htm#xtocid1083657
for more information.

First the ACL must be entered into the router. Using the cut and paste function, the entries can be copied onto the router quickly and easily. Enter global configuration mode and begin building the first list by typing:

```
ip access-list extended wan-in
```

Wan-in is the name of the access list. You will now be in access-list configuration mode. Cut and paste the below text (which should be in a text editor like notepad) into the router.

```
deny ip 192.169.252.0 0.0.1.255 any
deny ip 10.0.0.0 0.255.255.255 any
deny ip 172.16.0.0 0.15.255.255 any
deny ip 192.168.0.0 0.0.255.255 any
deny ip 127.0.0.0 0.255.255.255 any
deny tcp any any eq 23
deny tcp any any eq 22
deny tcp any any eq 21
deny tcp any any eq 139
deny tcp any any range 512 514
```

```
deny tcp any any eq 111
deny udp any any eq 111
deny tcp any any eq 2049
deny udp any any eq 2049
deny tcp any any eq 4045
deny udp any any eq 4045
deny tcp any any eq 135
deny udp any any eq 135
deny udp any any eq 137
deny udp any any eq 138
deny tcp any any eq 139
deny tcp any any eq 445
deny udp any any eq 445
deny tcp any any range 6000 6255
permit udp any host 192.169.252.10 eq 53
deny udp any any eq 53
deny tcp any any eq 53
deny tcp any any eq 389
deny udp any any eq 389
permit tcp any host 192.169.252.11 eq 25
deny tcp any any eq 25
deny tcp any any eq 109
deny tcp any any eq 110
deny tcp any any eq 143
permit tcp any host 192.169.252.12 eq 80
permit tcp any host 192.169.252.12 eq 443
deny tcp any any eq 80
deny tcp any any eq 443
deny tcp any any eq 8000
deny tcp any any eq 8080
deny tcp any any eq 8888
deny ip any any lt 21
deny tcp any any eq 37
deny udp any any eq 37
deny udp any any eq 69
deny tcp any any eq 79
deny tcp any any eq 119
deny tcp any any eq 123
deny tcp any any eq 515
deny udp any any eq 514
deny tcp any any eq 161
deny udp any any eq 161
deny tcp any any eq 162
deny udp any any eq 162
deny tcp any any eq 179
deny tcp any any eq 1080
```

```
deny icmp any any echo
permit ip any any
```

Exit out of access list configuration mode and build the next ACL by entering the following:

```
ip access-list extended wan-out
```

Now in access-list configuration mode, build the wan-out list by pasting in the following lines:

```
deny icmp any any echo-reply
deny icmp any any time-exceeded
deny icmp any any unreachable
permit ip any any
```

Exit out of access list configuration mode and build the next ACL by entering the following:

```
ip access-list extended perim-in
```

Now in access-list configuration mode, build the perim-in list by pasting in the following lines:

```
permit udp host 192.169.252.10 eq 53 host 192.169.253.10
permit tcp host 192.169.252.11 eq 25 host 192.169.253.11 established
permit tcp host 192.169.252.11 host 192.169.253.11 eq 25
deny ip any 192.169.253.0 0.0.0.255
permit ip 192.169.252.8 0.0.0.7 any
deny ip any any
```

Exit out of access list configuration mode. The ACLs must be applied to an interface before they are active. To apply the ACLs you will need to be in interface configuration mode.

For interface e0/0 enter the following:

```
ip access-group wan-in in
ip access-group wan-out out
```

For interface e0/1 enter the following:

```
ip access-group perim-in in
```

The ACLs are now operating on the router. In order to test the ACL, use the tool Firewalk (http://www.packetfactory.net/firewalk/) to see what ports are open. In our case we will have many more ports open than closed, but that is the nature of the security policy. To ensure Firewalk is able to receive the required time exceeded messages, test the filters without having wan-out active on the e0/0 interface. Firewalk will provide a list of open ports. Once the ports in the security policy are confirmed as closed, reapply the wan-out ACL to e0/0 and try Firewalk again. This time Firewalk should not be successful in identifying the open ports.

6. Conclusion

The filtering rules used to build the ACLs in this paper implement the required security policy. The internal network isn't too secure since all the ports not denied are permitted. It would be better to allow only what's needed and deny the rest; however, the rules in place are certainly better than no rules.