Global Information Assurance Certification Paper

Access Control Lists to Protect a Network from Worm/DoS Attacks

By Dennis Eck CCNA

December 4, 2003
GSEC Practical Assignment
Version 1.4, Option 1

Table of Contents

1. Abstract

Internet worms and Denial of Service (DoS) attacks have had a significant impact on businesses and governments in recent years. The damage caused by these attacks is measured in billions of dollars. Corporate productivity and government functioning are significantly impaired during these attacks.

Network security requires a Defense in Depth solution that is implemented at the client and server level as well as the network level. Solutions at the network level can include stateful firewalls, private virtual LAN's at the switch level, and packet filtering at the router level.

Five major Internet worms are reviewed in this paper: Code Red, Nimda, Slammer, Blaster, and Nachi. The network specific behavior of each virus is discussed along with research demonstrating that DoS attacks have the ability to completely overwhelm a network infrastructure.

The use of access control lists (ACL's) at the router level is a critical network security practice to safeguard a network infrastructure from worm/DoS attacks. ACL's should be implemented at several key points within a network. These locations include the network edges, including the Autonomous System (AS) boundaries and connection to the Internet, WAN links that connect geographically separate sites, and LAN access points to individual systems and end users.

This paper presents a variety of examples of ACL entries that can be used on a daily basis to protect a network from worm/DoS attacks. Examples of ACL entries are presented for general monitoring and blocking of malicious traffic, logging of potentially malicious traffic, blocking infected hosts, filtering out malicious traffic from mission critical systems, and an emergency stop ACL to block a significant worm/DoS attack.

Network administrators are encouraged to keep up to date with known vulnerabilities on the Internet and keep ACL's ready on their routers to implement at a moment's notice to protect their infrastructure.

2. Introduction

Network security has become serious business in the past few years. Internet worms and Denial of Service (DoS) attacks impact almost every user on the Internet, especially businesses and governments. While end users may experience slowness in connections or inaccessibility of certain websites, businesses and governments experience a significantly greater impact.

The financial impact sustained as a result of worm/DoS attacks is alarming. Code Red I and II cleanup costs totaled nearly $2.62 billion. The most expensive virus ever to hit was the Love Bug virus, which rang up $8.75 billion in damages by itself. Computer Economics, a research company that keeps a track of IT costs, published several

estimates on the economic damage caused by major computer viruses. The figure below provides their estimates for years 1995 to 2001. Data for the year 2002 was not available to the public as of this writing.

Analysis by Year Year	Worldwide Economic Impact ($ U.S. Billions)
2001	$13.2
2000	17.1
1999	12.1
1998	6.1
1997	3.3
1996	1.8
1995	0.5

Figure 1. Economic Impact of Malicious Code Attacks[1]

In addition to the financial impact, corporate productivity and a variety of critical government services have been significantly impacted by worms released on the Internet. The following list illustrates some of the documented damage:

- The Pentagon had to take down a number of its web sites
- The White House had to change its IP address
- Bank of America and Canadian Imperial Bank reported that many customers could not withdraw money from ATMs
- Internet congestion prevented consumers from contacting Microsoft over the Internet to download patches
- Millions of South Korean users lost Internet access when Korea Telecom Freetel and SK Telecom service failed
- Networks at the U.S. departments of State, Agriculture, Commerce were disrupted
- Some Associated Press news services and several newspapers were temporarily interrupted
- Trading volume at the Korea Stock Exchange fell to a 13-month low as a result of investors avoiding submission of orders to buy
- Continental Airlines reported widespread computer problems including delays at several major airports
- Operations were disrupted within a number of high-profile companies such as Qwest Communications, AT&T, FedEx, and Intel.

[1] Waite, "Malicious Code Attacks Had $13.2 Billion Economic Impact in 2001."

The solution to protect networks in the current inter-networked world requires a Defense-In-Depth approach. This approach uses a layered design that involves multiple systems, devices, and strategies that function together to minimize security holes and vulnerabilities. At the server and PC level, most major vendors have defined a common set of defense measures to prevent infection and hacker attacks. The following list outlines these standard defense measures:

- Install antivirus software and keep scan engine and virus definition files current
- Keep OS and software patch levels current
- Enable a firewall that is native to the OS or install a personal third party firewall
- Disable or remove any unnecessary services
- Filter or delete email that contains .vbs, .bat, .exe, .pif or .scr file attachments since these attachments are known to be used as a vehicle to spread viruses
- Instruct employees not to open emails with attachments from unknown sources
- Implement a strong password policy

In addition to the security measures in place at the client and server level, there are several defense methods that can be employed at the network level to protect against worms/DoS attacks. These security measures can include stateful firewalls, separating traffic at the switch level into private virtual LANs (VLAN), and packet filtering at the router level. This paper will focus on implementing access control lists on Cisco routers control worm/DoS attacks.

3. Major Internet Worm Attacks

There have been a handful of major Internet worms released on the Internet in the past few years. These worms created such a volume of traffic that network pipes became severely congested and network devices were overloaded to the point where they were unable to forward legitimate traffic, or the network devices may have literally crashed under the load. The result of these worms on the Internet was a Denial of Service attack against many ISP's and business networks, even though this may not have been the original intent of the attack.

In July 2001, the Code Red Worm was released on the Internet. According to data collected by the Cooperative Association for Internet Data Analysis (CAIDA):

> …more than 359,000 computers connected to the Internet were infected with the Code-Red worm in less than 14 hours… Even without being optimized for spread of infection, Code-Red infection rates peaked at over 2,000 hosts per minute.[2]

Code Red affected Microsoft Index Server 2.0 and the Windows 2000 Indexing service on computers running IIS 4.0 and 5.0 Web servers. The worm sent its code as an HTTP request on port 80 which exploited a known buffer-overflow vulnerability. The worm also attempted a Denial of Service (DoS) against www.whitehouse.gov (198.137.240.91) if the date was between the 20th and 28th of the month.

[2] Moore, "Code-Red: a case study." p.1.

Code Red used a random number generator to get new IP addresses to attack. The initial revision of Code Red hit the same machines over and over again which limited the worm's ability to spread. Code Red II used a better random number generator to create more target IP addresses by keeping the network portion of the IP address, and then choosing a random host portion of the IP address. This allows the worm to spread itself faster within the same network.

A few months after Code Red was released, the Nimda worm surfaced. It took advantage of some similar vulnerabilities as Code Red, however, it was a hybrid attack that contained both worm and virus characteristics. As a more advanced attack, it could infect more systems and could infect systems in multiple ways. Nimda could infect any computer running Microsoft Windows software by exploiting a flaw in Outlook Express and known vulnerabilities in Microsoft's Internet Information Services software (IIS) 4 or 5, including the security hole left by Code Red II.

Nimda used randomly generated IP addresses to target vulnerable IIS servers using TCP 80 (HTTP). It copied itself to the vulnerable web server as a DLL using TFTP (UDP port 69), then created a listening port ready to transfer the copy of the worm. The transfer of the worm used NetBIOS TCP ports 137-139 or TCP port 445. Nimda then searched for all open network shares using Network Neighborhood. Nimda also used TCP port 25 (SMTP) for sending email to other systems using addresses taken from the infected system.

The Slammer worm was released in January 2003. The primary impact of Slammer was a consumption of network bandwidth. It infected systems at a rate not seen before and saturated networks quickly. The worm created a Denial of Service attack due to the large number of packets it sends. In some cases, networks experienced 100% packet loss.

Slammer targeted systems running Microsoft SQL Server 2000 and Microsoft Desktop Engine (MSDE) 2000. It took advantage of a buffer overflow vulnerability that allowed a part of system memory to be overwritten, then ran in the same security context as the SQL Server service. Slammer used UDP for connection setup so it did not have the overhead of the connection setup and management required by TCP-based services. Code Red and Nimda took advantage of flaws in TCP-based services, which required a full three-way handshake before exchanging data. Slammer flooded the network by continuously sent traffic consisting of 376 byte packets to UDP port 1434 of randomly generated IP addresses.

In August 2003, the Blaster worm hit the Internet, targeting Microsoft Windows 2000 and XP systems in an attempt to take advantage of a recent published Microsoft Windows RPC DCOM vulnerability. The Distributed Component Object Model (DCOM) allows Microsoft software to communicate. This includes communication with Internet protocols such as HTTP. A flaw in the RPC code was exploited to cause a buffer overflow and the RPC service would fail. This allowed an attacker to run code with

System privileges which included the ability to install programs, change data and create accounts with administrator rights.

Blaster used several ports on compromised systems as listening ports or to execute commands. It scanned a random IP range looking for systems to infect on TCP port 135, which is the port that the RPC process listens on. The traffic created by Blaster saturated subnets with port 135 requests. TCP port 4444 was also a listening port that allowed an attacker to issue remote commands through a hidden shell process. UDP port 69 listened for requests to transmit the virus from computers that were compromised by the RPC DCOM vulnerability. The worm also sent 40 byte HTTP packets on port 80 to windowsupdate.com at the rate of 50 packets per second. If it was unable to find a DNS entry for windowsupdate.com, Blaster used a broadcast address of 255.255.255.255.

About one week after the Blaster worm was released, the Nachi worm surfaced with some interesting features for a virus or worm. While it took advantage of the same RPC DCOM vulnerability that Blaster did, it attempted to remove Blaster, then download and install the RPC DCOM patch from Microsoft.

Nachi selected potential victims either by random IP addresses, or it searched starting with an IP address range using the same first two octets of the infected system's IP address. It sent an ICMP echo request (PING) to find active machines on the network. The most visible network activity from Nachi was high volumes of 92 byte ICMP echo request (PING) packets.

Once it found a machine, it either used TCP port 135 to exploit RPC DCOM vulnerability, or it used TCP port 80 to exploit the WebDav vulnerability. World Wide Web Distributed Authoring and Versioning (WebDAV) is implemented when IIS is installed. If an excessive amount of data was sent to WebDAV, the data was forwarded to the ntdll.dll, which failed to perform proper checks on the data and allowed a buffer to be overrun. The resulting buffer overflow allowed for execution of code in the IIS service, which runs with System level privileges.

After it invades a system, a remote shell was created, and the system reconnected to the attacking system using a TCP port between 666 and 765. Commonly, it used port 707. The port was used to establish a control channel to issue commands for downloading svchost.exe and dllhost.exe from an infected system. The actual downloads occurred using TFTP, which runs on UDP port 69.

4. Network Impact

Denial-of-service (DoS) attacks, such as those Internet worms discussed above, present a significant problem for networks. They can shut an organization off from the Internet or seriously disrupt both LAN access and site-to-site WAN communications. The CERT Coordination Center has reported that DoS attacks are increasingly being directed against network components:

> Windows end-users and Internet routing technology have both become more frequent targets of intruder activity…Bandwidth, processing power, and storage capacities are all common targets for DoS attacks…the most common DoS attack type reported to the CERT/CC involves sending a large number of packets to a destination causing excessive amounts of endpoint, and possibly transit, network bandwidth to be consumed. Such attacks are commonly referred to as packet flooding attacks. [3]

The severe demand placed on network equipment during a DoS attack was studied by David Moore with the Cooperative Association for Internet Data Analysis (CAIDA). Moore's research quantified the amount of traffic that can be seen during a DoS attack:

> Half of the uniform random attack events have a packet rate greater than 250, whereas half of all attack events have a packet rate greater than 350. The fastest uniform random event is over 517,000 packets per second, whereas the fastest overall event is over 679,000 packets per second…In our trace, 38% of uniform random attack events and 46% of all attack events had an estimated rate of 500 packets per second or higher. [4]

The increased traffic load placed on networking equipment during a DoS attack can severely impact the device's ability to handle legitimate traffic. The common conditions seen on a router during a DoS attack include:

- High CPU usage
- Routing protocol drops
- Packet loss
- ARP storms
- Excessive memory use.

The ultimate result is that the router may reload itself, and may continue to reload itself until the DoS attack subsides.

Additionally, firewalls can be overwhelmed by the amount of traffic they receive. Moore's research with CAIDA further demonstrates that DoS attacks have the ability to compromise firewalls:

---

[3] Houle, "Trends in Denial of Service Attack Technology." p.1-3
[4] Moore, "Inferring Internet Denial-of-Service Activity." p.8.

> Recent experiments with SYN attacks on commercial platforms show that an attack rate of only 500 SYN packets per second is enough to overwhelm a server…The same experiments show that even with a specialized firewall designed to resist SYN floods, a server can be disabled by a flood of 14,000 packets per second. In our data, 0.3% of the uniform random attacks and 2.4% of all attack events would still compromise these attack-resistant firewalls. We conclude that the majority of the attacks that we have monitored are fast enough to overwhelm commodity solutions, and a small fraction are fast enough to overwhelm even optimized countermeasures. [5]

In addition to the impact to a particular network device, an attack against one site may also affect network resources that serve multiple sites. A downstream site may experience increased network latency, packet loss, backed up mail queues, or possibly a complete outage. Even though the downstream site may not have been the original target of the attack, the upstream connections and routers may not have the ability to handle legitimate traffic destined for that site. The network impact of a DoS attack grows significantly if there are a large number of infected systems that have a large amount of outbound bandwidth.

5. Ports to Monitor or Block

There are several common ports with known vulnerabilities that have been targets for worm attacks in recent years. These ports will always require monitoring, some will need traffic filtering, and some will need to be blocked completely at certain points within the network.

TFTP (Trivial File Transport Protocol, UDP port 69) can be used by infected systems to transfer files from an infected system to a newly exploited machine. Examples include the msblast executable from the Blaster worm and svchost and dllhost executables from Nachi. The spread of these worms can be prevented by blocking UDP port 69. However, installation of software images or configurations to networked devices will be impacted, since this is a primary use of TFTP.

HTTP (HyperText Transport Protocol, TCP port 80) is primarily used for access to the Worldwide Web. The HTTP daemon on a web server uses port 80 to listen for traffic from a web client. Multiple worms attempt to exploit various known vulnerabilities in Microsoft's Internet Information Services (IIS) to create buffer overflows on vulnerable systems and gain system level access. Blocking port 80 can prevent initial infections and their spread, but it will block access to a variety of web-based applications.

The Microsoft RPC protocol uses TCP port 135 to allow RPC clients to determine the port number currently assigned to a particular RPC service. When trying to connect to a service, the client goes through the Endpoint Mapper to discover where the service is located. Applications such as WINS, DHCP, and DNS depend on the RPC protocol. Port 135 is also essential to the functionality of Active Directory and Microsoft Exchange

[5] Moore, "Inferring Internet Denial-of-Service Activity." p.8.

mail servers. This port is where the MS RPC DCOM vulnerability is initially exploited to start the sequence of events that fully infects a machine with Blaster or Nachi. Blocking port 135 can prevent infections by these worms, but may impact email and file access within the local network.

NetBIOS functions (TCP and or UDP ports 137-139) have known vulnerabilities that leave hosts open to exploitation. There is no need for these ports to be directly accessible from the Internet. These ports are needed for file sharing on the internal network, but they should be blocked at both the ingress and egress points at edge networks to prevent remote exploitation.

Microsoft Domain Service uses TCP port 445 uses for SMB (Server Message Block) over TCP. Similar to NetBIOS services, this protocol is also used for file sharing in Windows NT/2000/XP. Prior to Windows 2000/XP, Windows NT ran NetBIOS over TCP/IP (ports 137-139). Windows 2000/XP provides the ability to run SMB directly over TCP/IP without using NetBIOS. TCP port 445 is being targeted by a variety of recent worm attacks.

TCP port 707 is used by the Nachi worm to open a control channel between an infected system and a vulnerable system. This channel is used to pass commands between the systems to download the svchost and dllhost executables from the infected system. Blocking port 707 can prevent the spread of Nachi by interfering with the worm's ability to communicate between infected and vulnerable systems.

UDP port 1434 is used by Microsoft SQL Server as a listener service that allows a client to query the SQL server for a list of named instances and related network configuration information. After the client establishes a TCP/IP connection, the client opens a source port and sends traffic to a destination port, which is TCP port 1433 by default. The source port on the client may vary, but it is greater than 1024. Firewalls or perimeter routers should be configured to block unsolicited traffic to and from UDP port 1434. Remote connections to local SQL databases should be blocked unless the connection attempts originate from trusted hosts and networks. Blocking UDP port 1434 in this manner will prevent the spread of any Slammer infections.

TCP port 4444 is used for Kerberos authentication and Oracle9i communication. The Blaster worm uses this port to open a command shell on the infected machine so that it can be remotely controlled. Blocking this port will prevent a system from being used to launch attacks on other systems, however it can disrupt Kerberos authentication or Oracle9i communications.

ICMP protocol type 8 is the PING utility used for connectivity testing within a network. The ICMP protocol is quite different from the TCP/IP and UDP/IP protocols. No source and destination ports are included in its packets, so the normal packet-filtering rules for TCP/IP and UDP/IP are different. Blocking this protocol can prevent the spreading of worms such as Nachi that uses ICMP echo requests, but it will create problems with network diagnostics.

6. ACL Basics

Access Control Lists (ACL's) are used in routers to identify, manage, and filter traffic in a network. They filter the packet flow coming in or going out of router interfaces to control network traffic and restrict network use to certain users or devices. ACL's act on traffic that is going through the router but do not act on packets that originate from the router itself.

Standard ACL's are used to control network access by specific hosts or networks. These ACL's control access based on the source address of the IP packets. Standard ACL's commonly use identification numbers from 1 to 99, but they can also use names instead of numbers. When ACL's use names instead of numbers, they are referred to as Named ACL's. The basic syntax for a standard access control list is:

access-list <1 to 99> <permit or deny> <source IP address> <mask>

Extended ACL's are used to filter specific types of traffic to and from specific locations. These ACL's control traffic by protocol, source address, and destination address of the IP packets. Extended ACL's use identification numbers from 101 to 199. Like Standard ACL's, Extended ACL's can also use names instead of numbers. The basic syntax for an extended access control list is:

access-list <100 to 199> <permit or deny> <protocol> <source IP address> <source mask> <destination IP address> <destination mask>

Additional information can be added to an extended access list to filter traffic using specific ports and service, such as FTP, telnet, etc. Operators such as eq (equals), gt (greater than), and range (of port numbers) are added to the syntax to determine how the traffic will be filtered in relation to the port or service. The syntax for an extended ACL using port or service is:

access-list <100 to 199> <permit or deny> <protocol> <source IP address> <source mask> <operator> <port or service> <destination IP address> <destination mask> <operator> <port or service>

Remarks can be added before or after an entry in an access control list to describe the function of the ACL entry. The basic syntax for an ACL remark is:

access-list <1 to 99> remark <purpose of the deny or permit statement>

Masks used in ACL statements are known as inverse or wildcard masks. These masks are similar to standard IP address subnet masks, but they are actually written in a reverse format. ACL masks specify what traffic should be permitted and denied. When a mask is converted into binary ones and zeros, it determines which bits of the IP address in the ACL should be used in processing the traffic. A zero in the inverse mask indicates

that the bits in the IP address must match; a one indicates that the address bit will be ignored.

For example, a wildcard mask of 0.0.0.0 would be used to identify a specific host since all bits in the mask will be matched with the specified IP address. The keyword *host* is now commonly substituted by the router in front of the IP address when a wildcard mask of 0.0.0.0 is used. Wildcard masks of 0.255.255.255, 0.0.255.255, or 0.0.0.255 would be examples of wildcard masks used to filter standard class A, B, or C networks in an ACL. A source address and wildcard mask of 0.0.0.0 255.255.255.255 will include all hosts and networks and can be specified as *any* in the ACL syntax.

The most frequently hit entries should appear first in an access list. This saves the router from additional processing that occurs as a result of checking unnecessary lines in an ACL. There is an implicit deny statement built into the end of all access lists even if it is not listed. Therefore, an ACL must have at least one permit statement, or it will block all traffic.

ACL's can be added to a router's configuration but will not have any impact until applied to a specific interface. After an access list is applied to a router interface, the router will check all packets coming into that interface and compare them in the order that the entries occur in the access list. The router will check all entries in the access list until it finds a match. If the packet does not match an entry, it will be discarded based on the implicit deny statement.

Each interface can have only one access list applied for each protocol (e.g. IP, IPX) and for each direction- in and out. Input access lists save processing on the router since the router does not have to perform routing table lookups to find which interface handles the traffic for the destination address. For best efficiency in a routed network, ACL's should be applied to the interface closest to the source of the traffic. The figure below from Cisco illustrates where the ACL should be applied (Router A, interface E0) in relation to the traffic source and destination.

Figure 2. Location to Apply ACL's for Best Efficiency [6]

Packet filtering using ACL's should be done as early as possible in a network. Discarding traffic as close to the source as possible has several benefits:

- The impact of any DoS attack is limited to the perimeters of the network
- Systems are protected from being overloaded
- Attackers are restricted from using internal systems to attack other sites

[6] "Configuring IP Access Lists."

- Regular operations can continue uninterrupted
- Network resources are not tied up processing undesirable traffic.

7. ACL Development and Implementation

ACL's should be developed proactively in anticipation of a potential attack rather than trying to develop completely new ACL's during a crisis. Network security administrators should always be aware of the latest reported vulnerabilities and viruses so that they can effectively develop and update ACL's to protect their networks before an attack begins. The ACL's can remain in the router's configuration without impacting the router's performance or acting to filter any packets. Until the ACL's are actually applied to a router interface, they have no effect.

The ACL entry examples listed in this section are presented as a starting point to protect a network from worm/DoS attacks. In addition to these examples, network administrators will need to develop more comprehensive access control lists to secure their network. More comprehensive ACL's may include entries for anti-spoofing addresses, infrastructure protection to control access to servers and network devices, and specific permit and deny statements related to internal applications and hosts.

There should be access lists in place at key points within a network to monitor potentially undesirable traffic and filter known malicious traffic. Access control lists should be in place at the ingress and egress points of a network to provide the first line of defense against invalid or malicious traffic. This includes the boundaries between network Autonomous Systems (AS) and a company's connection to the Internet.

In an effort to protect a network from malicious traffic originating from external sources, ACL's at network ingress points can control access to infrastructure devices (routers, switches, and servers) and control transit traffic bound for client systems. Incoming connection attempts from external devices should be blocked to hosts or networking devices that are providing a service to the internal network only. The attack can blocked at the network edge, and the traffic will be dropped by the ingress interfaces into the network.

Access control lists should also in place at the egress points to block devices that do not need outbound Internet access. This includes many networking devices and servers that only provide services to the internal network. If an internal device is infected, attempts to infect devices on external networks will be discarded at the perimeter of the network. Egress filtering can also prevent internal infected systems from flooding the WAN and Internet links which will severely degrade the successful transmission of legitimate inbound or outbound traffic.

ACL's on a border router (ingress or egress points) should have anti-spoofing entries and block special-use addresses. This will help deny illegitimate source and private IP address ranges from sending packets into the network from an external source. Common ACL entries on a border router would include:

```
access-list XXX deny ip host 0.0.0.0 any
access-list XXX remark illegitimate source address

access-list XXX deny ip 127.0.0.0 0.255.255.255 any
access-list XXX remark illegitimate source address

access-list XXX deny ip 192.0.2.0 0.0.0.255 any
access-list XXX remark reserved for Test-Net

access-list XXX deny ip 224.0.0.0 31.255.255.255 any
access-list XXX remark address range for multicast use only

access-list XXX deny ip 10.0.0.0 0.255.255.255 any
access-list XXX remark non-routable address range for private internal use only

access-list XXX deny ip 172.16.0.0 0.15.255.255 any
access-list XXX remark non-routable address range for private internal use only

access-list XXX deny ip 192.168.0.0 0.0.255.255 any
access-list XXX remark non-routable address range for private internal use only
```

Additionally, there should be entries to protect against known vulnerabilities and block known worms/DoS attacks. For example, NetBIOS should not need access beyond the local area network (LAN). Since the ports associated with NetBIOS have been a common target for worms, they should be filtered out at the Internet and WAN levels. If there are reasons why NetBIOS ports or other known vulnerable ports (e.g. TFTP) should be accessible, there should be specific permit statements allowing traffic only from trusted hosts and networks. The following entries can also be added to the ACL on a border router to protect vulnerable ports and block known worms:

```
access-list XXX deny tcp any eq 707 any
access-list XXX remark blocks Nachi control channel

access-list XXX deny tcp any any eq 707
access-list XXX remark blocks Nachi control channel

access-list XXX deny udp any gt 1024 any eq 1434
access-list XXX remark blocks unsolicited SQL traffic to prevent Slammer

access-list XXX permit tcp any any eq www
access-list XXX remark monitors port 80 traffic to check for Code Red, Nimda, etc.
```

```
access-list XXX deny tcp any any eq 135
access-list XXX remark blocks traffic potentially related to RPC DCOM vulnerability

access-list XXX deny udp any any range 137-139
access-list XXX remark blocks NetBIOS traffic

access-list XXX deny tcp any any eq range 137-139
access-list XXX remark monitors NetBIOS traffic

access-list XXX deny tcp any any eq 445
access-list XXX remark blocks MS Domain Service traffic

access-list XXX deny udp any any eq 69
access-list XXX remark blocks TFTP traffic, possible Blaster or Nachi traffic
```

Internal address space should never be the source of packets from outside the WAN. There should also be entries in the ACL on a border router to deny incoming packets with source IP addresses from the address range of the internal network. The syntax for this ACL entry would be:

```
access-list XXX deny ip <internal address block> <mask> any
```

Access control lists at the DMZ boundary should allow incoming HTTP requests to servers such as company E-commerce and web servers since these services will need to be available to the public. These are the only devices that should generally be directly accessible from the Internet. The internal network behind the DMZ boundary should never directly accessible from the Internet, but the internal network must have the ability to reach Internet sites. The ACL's should allow specifically permitted traffic from internal users to and from the DMZ and Internet. Any unauthorized traffic should be dropped on the ingress interfaces to the DMZ. Access control lists should also be available at the DMZ to be implemented whenever necessary to block potentially infected public servers from infecting private internal servers.

There should be several different access lists available at both the WAN and LAN level that can be used either on a daily basis or as needed to stop a potential infection or undesirable traffic. This will prevent vulnerable or compromised devices from infecting systems at other sites or on other subnets.

The first ACL should be a general basic monitoring and blocking ACL that monitors or blocks known worms/DoS attacks. On a WAN router, NetBIOS and other common LAN traffic would generally be denied. There should be specific permit statements in the ACL to allow certain hosts and subnets access to these vulnerable ports on an as needed basis. Additionally, it can be wise to use specific permit statements for access to port 80 (WWW) and port 69 (TFTP) based on trusted internal network addresses rather than having a blanket permit statement allowing access to port 80. Entries for this type of WAN ACL could include:

access-list XXX deny tcp any eq 707 any
access-list XXX remark blocks Nachi control channel

access-list XXX deny tcp any any eq 707
access-list XXX remark blocks Nachi control channel

access-list XXX deny udp any gt 1024 any eq 1434
access-list XXX remark blocks unsolicited SQL traffic to prevent Slammer

access-list XXX deny tcp any any eq 4444
access-list XXX remark blocks Blaster attempts to setup remote control on infected system

access-list XXX permit tcp any any eq www
access-list XXX remark monitors port 80 traffic to check for Code Red, Nimda, etc.

access-list XXX deny tcp any any eq 135
access-list XXX remark blocks traffic potentially related to RPC DCOM vulnerability

access-list XXX deny udp any any range 137-139
access-list XXX remark blocks NetBIOS traffic

access-list XXX deny tcp any any eq range 137-139
access-list XXX remark monitors NetBIOS traffic

access-list XXX deny tcp any any eq 445
access-list XXX remark blocks MS Domain Service traffic

access-list XXX deny udp any any eq 69
access-list XXX remark blocks TFTP traffic, possible Blaster or Nachi traffic

On a LAN router the above statement for ports 135, 137-139, 445, and 69 would be deleted or changed to permit statements since services on these ports would need access between LAN segments.

The number of hits on each entry in an access list can be viewed to quantify the level of traffic filtered by the router. This data can be useful in several regards. It can provide a baseline of routine traffic within the network. It can help identify a new attack, quantify the rate of infection for a current attack, or determine when an attack subsides. To view the hits against an ACL, the *show access-list <ACL number>* command will display a similar output to the following example:

Extended IP access list <ACL number>
deny tcp any eq 707 any(# matches)
deny tcp any any eq 707(# matches)
deny udp any gt 1024 any eq 1434(# matches)

permit tcp any any eq www(# matches)
deny tcp any any eq 135(# matches)
deny udp any any range 137-139(# matches)
deny tcp any any range 137-139(# matches)
deny tcp any any eq 445(# matches)
permit ip any any(# matches)

The hits on an ACL entry can also be viewed on the router as they occur using a logging ACL. The logging ACL would have the same entries as the general monitoring and blocking ACL, but the log keyword can be added to the entries to provide the ability to monitor the terminal output. Using the *terminal monitor* command, an administrator can observe the hits on a specific ACL entry as they occur. When applied to an interface, excessive hits to an ACL entry using the log keyword will generate a high number of log entries and increase CPU utilization. Therefore, this ACL should only be applied to an interface for a few minutes at a time.

As an example, an administrator could log port 80 traffic to check for acceptable HTTP traffic versus Code Red or Nimda. The administrator would watch for a high number of single packets being generated from the same source IP address to various destination IP addresses. Here is an example of an ACL entry that could be used to check port 80 traffic for Code Red or Nimda infections:

access-list XXX permit tcp any any eq www log
access-list XXX remark logs port 80 traffic to check for Code Red, Nimda, etc.

The router configuration should also include a standard access list to completely block network access by potentially infected host(s) or subnet(s). This ACL would be applied to an Ethernet or Fast Ethernet interface on a LAN router until the host or subnet can be blocked at the switch level by disabling the necessary switch port(s). Below is an example of the syntax for an IP access-list to block infected host(s):

access-list XX deny <infected host address> 0.0.0.0 (This line should be repeated for each infected host.)
access-list XX permit any

There will be circumstances when a specific host or hosts may be infected, but the host serves a mission critical function and cannot be removed from the network immediately. An extended ACL should be available on the router to block the malicious traffic from the infected host(s). This ACL would also be applied to Ethernet or Fast Ethernet interface on the router, but the device remains available to serve other requests. An example of ACL to block port 80 traffic from an infected host(s) is:

access-list XXX deny tcp host <infected host address> any eq 80 (This line should be repeated for each infected host.)
access-list XXX permit ip any any

Finally, there should be an extended ACL available on a router as an emergency stop in case of a virus outbreak or DoS attack. This ACL would be similar to the general monitoring and blocking ACL that is applied to most router interfaces. However, permit statements in place for monitoring can be changed to deny statements to block suspected worm/DoS traffic. Since this ACL could block virtually all traffic of a particular type through a network, it should only be applied in critical situations to prevent rapid spread of a virus.

TCP port 445 has been a target of recent attacks. Here is an example where the syntax of one entry in the general monitoring and blocking ACL has been modified to block a potential worm infection:

```
access-list XXX deny tcp any any eq 445
access-list XXX remark blocks all traffic destined for TCP port 445
```

8. Finding and Blocking Infected Hosts with ACL's

When a worm or DoS attack is suspected, it is critical to quickly isolate or remove the infected host(s) or network(s). It would be better to spend a few minutes to block a few infected hosts or a subnet than spend hundreds of man hours battling a major outbreak. Some worm/DoS attacks have the ability to propagate so quickly (e.g. Slammer) that it would be impossible to isolate the infected hosts with a blocking ACL.

Here are some steps that can be taken using a logging ACL to trace down infected host(s):

1. Apply a logging ACL to each active Ethernet or FastEthernet interface. Make sure that the ACL is applied in the same direction (in/out) as the general monitoring and blocking ACL was applied.
2. Run the *terminal monitor* command to monitor the hits on the ACL for a short period of time.
3. Capture the data and output to a text file.
4. After a few minutes, cancel the terminal monitor using the *terminal no monitor* command to stop output to the screen.
5. Reapply the general monitoring and blocking ACL to each active Ethernet or FastEthernet interface. Make sure that ACL is reapplied in the same direction (in/out) as it was originally.
6. Analyze the captured data for packet transmissions that are consistent with the suspected worm or DoS attack. Identify the potentially infected source IP addresses.
7. Use the *show IP route* <source IP address> command on the router to identify the interface where the potentially infected host is connected.
8. Modify the blocking ACL (either a complete block or blocking malicious traffic) by adding the IP addresses for each potentially infected host or subnet.
9. Apply the blocking ACL inbound to each Ethernet or FastEthernet interface to the network that has a route to the infected host(s) or subnet(s).

Ideally, it is best to block infected hosts by shutting down the related port on the nearest LAN switch. However, there are several cases where this is not immediately available. Examples include:

- The host may serve a mission critical function
- The host may be connected to a switch in a lower layer of a multilayered switched environment
- The host may be connected to an unmanaged hub

9. Conclusion

The use of access control lists to filter traffic within a routed network is a critical network security practice. ACL's provide network administrators with the ability to monitor vulnerable ports and block known malicious traffic at key points within a network. The access control lists in place at the ingress and egress points of a network are a key part of the first line of defense. The filtering strategy in place at the network edges reduces many of the risks associated with direct network attacks.

Access control lists in place at the WAN and LAN level will guard against compromised or infected systems from attacking vulnerable systems on other subnets or at other sites. There should be several access control lists in the router's configuration for use on a daily basis, or waiting to be used to block infected hosts or malicious traffic.

Network security administrators should be aware of the current vulnerabilities so that ACL's can be updated and waiting in a router's configuration before an actual attack begins. This practice can help isolate an attack quickly and save hundreds of man hours that would be required to battle a full scale outbreak.

10. References

Knowles, Douglas. "W32.SQLExp.Worm." Symantec Security Response. 04 February 2003. URL: http://securityresponse.symantec.com/avcenter/venc/data/ w32.sqlexp. worm.html (29 July 2003)

Author not listed. "MS SQL Worm Mitigation Recommendations." Cisco Security Notice. 19 May 2003. URL: http://www.cisco.com/warp/public/707/cisco-sn-20030125-worm. shtml (29 July 2003)

Chien, Eric. "W32.Nimda.A@mm." Symantec Security Response. 11 July 2003. URL: http://securityresponse.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html (30 July 2003)

Author not listed. "How to Protect Your Network Against the Nimda Virus." Cisco Systems Tech Notes. 23 April 2003. URL: http://www.cisco.com/en/US/products/sw/ iosswrel/ps1835/products_tech_note09186a0080110d17.shtml. (30 July 2003)

Chien, Eric. "CodeRed Worm." Symantec Security Response. 29 July 2003. URL: http://securityresponse.symantec.com/avcenter/venc/data/codered.worm.html. (30 July 2003)

Author not listed. "Configuring IP Access Lists." Cisco Systems Tech Notes. June 13, 2003. URL: http://www.cisco.com/en/US/products/sw/secursw/ps1018/ products_ tech_note09186a00800a5b9a.shtml. (30 July 2003)

Householder, Allen et al. "Managing the Threat of Denial-of-Service Attacks." CERT® Coordination Center. v10.0 October 2001. URL: http://www.cert.org/ archive/pdf/Managing_DoS.pdf. (1 August 2003)

McQuerry, Steve. Interconnecting Cisco Network Devices. Indianapolis, IN. Cisco Press, July 2001. 297-303.

Author not listed. "SAFE Code-Red Attack Mitigation." Cisco Systems White Papers. 1 August 2003. URL: http://www.cisco.com/application/pdf/en/us/guest/netsol/ns128 /c654/cdccont_0900aecd800b05ab.pdf (24 September 2003)

Author not listed. "SAFE Nimda Attack Mitigation." Cisco Systems White Papers. 1 August 2003. URL: http://www.cisco.com/application/pdf/en/us/guest/netsol/ns128/ c654/ccmigration_09186a008009c8ae.pdf (24 September 2003)

Author not listed. "SAFE SQL Slammer Worm Attack Mitigation." Cisco Systems White Papers. 1 August 2003. URL: http://www.cisco.com/application/pdf/en/us/guest/netsol/ ns128/c654/cdccont_0900aecd800b05b5.pdf (24 September 2003)

Waite, Beverly. “Malicious Code Attacks Had $13.2 Billion Economic Impact in 2001.” Computer Economics. 4 January 2002. URL: http://www.computereconomics.com/article.cfm?id=133. (5 August 2003)

Author not listed. “SAFE RPC DCOM/W32/Blaster Attack Mitigation.” Cisco Systems White Papers. 29 August 2003 URL: http://www.cisco.com/en/US/netsol/ns110/ns170/ns171/ns128/networking_solutions_white_paper09186a00801b2391.shtml (25 September 2003)

Author not listed. “Cisco Security Notice: W32.BLASTER Worm Mitigation Recommendations.” Cisco Systems Tech Notes. 21 August 2003 URL: http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_tech_note09186a00801aedd6.shtml (25 September 2003)

Author not listed. “Cisco Security Notice: Nachi Worm Mitigation Recommendations.” Cisco Systems Tech Notes. 13 September 2003. URL: http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_tech_note09186a00801b143a.shtml. (25 September 2003)

Knowles, Douglas et al. “W32.Blaster.Worm.” Symantec Security Response. 29 August 2003. URL: http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html (25 September 2003)

Nahorney, Benjamin et al. “W32.Welchia.Worm.” Symantec Security Response. 24 September 2003. URL: http://securityresponse.symantec.com/avcenter/venc/data/w32.welchia.worm.html (25 September 2003)

Author not listed. “Microsoft Windows 2000 WebDAV / ntdll.dll Buffer Overflow Vulnerability.” Symantec Security Response. Date/revision not listed. URL: http://securityresponse.symantec.com/avcenter/security/Content/3.17.2003.html (26 September 2003)

Moore, David et al. “Inferring Internet Denial-of-Service Activity.” The Cooperative Association for Internet Data Analysis (CAIDA). 13 September 2002 URL: http://www.caida.org/outreach/papers/2001/BackScatter/usenixsecurity01.pdf (26 September 2003)

Moore, David et al. “Code-Red: a case study.” The Cooperative Association for Internet Data Analysis (CAIDA). 3 December 2002 URL: http://www.caida.org/outreach/papers/2002/codered/codered.pdf (26 September 2003)

Houle, Kevin J. and Weaver, George M. “Trends in Denial of Service Attack Technology.” CERT Coordination Center. October 2001 URL: http://www.cert.org/archive/pdf/DoS_trends.pdf. (26 September 2003)

Last Updated: May 28th, 2017

Upcoming Training

SANS Stockholm 2017	Stockholm, Sweden	May 29, 2017 - Jun 03, 2017	Live Event
SANS San Francisco Summer 2017	San Francisco, CA	Jun 05, 2017 - Jun 10, 2017	Live Event
Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
Community SANS Ottawa SEC401	Ottawa, ON	Jun 05, 2017 - Jun 10, 2017	Community SANS
SANS Rocky Mountain 2017	Denver, CO	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Charlotte 2017	Charlotte, NC	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Rocky Mountain 2017 - SEC401: Security Essentials Bootcamp Style	Denver, CO	Jun 12, 2017 - Jun 17, 2017	vLive
Community SANS Portland SEC401	Portland, OR	Jun 12, 2017 - Jun 17, 2017	Community SANS
SANS Secure Europe 2017	Amsterdam, Netherlands	Jun 12, 2017 - Jun 20, 2017	Live Event
SANS Minneapolis 2017	Minneapolis, MN	Jun 19, 2017 - Jun 24, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS Cyber Defence Canberra 2017	Canberra, Australia	Jun 26, 2017 - Jul 08, 2017	Live Event
SANS Paris 2017	Paris, France	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
Cyber Defence Japan 2017	Tokyo, Japan	Jul 05, 2017 - Jul 15, 2017	Live Event
SANS Cyber Defence Singapore 2017	Singapore, Singapore	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Minneapolis SEC401	Minneapolis, MN	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS Los Angeles - Long Beach 2017	Long Beach, CA	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Phoenix SEC401	Phoenix, AZ	Jul 10, 2017 - Jul 15, 2017	Community SANS
SANS Munich Summer 2017	Munich, Germany	Jul 10, 2017 - Jul 15, 2017	Live Event
Mentor Session - SEC401	Macon, GA	Jul 12, 2017 - Aug 23, 2017	Mentor
Mentor Session - SEC401	Ventura, CA	Jul 12, 2017 - Sep 13, 2017	Mentor
Community SANS Atlanta SEC401	Atlanta, GA	Jul 17, 2017 - Jul 22, 2017	Community SANS
Community SANS Colorado Springs SEC401	Colorado Springs, CO	Jul 17, 2017 - Jul 22, 2017	Community SANS
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC401: Security Essentials Bootcamp Style	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
Community SANS Charleston SEC401	Charleston, SC	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Fort Lauderdale SEC401	Fort Lauderdale, FL	Jul 31, 2017 - Aug 05, 2017	Community SANS
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event