Global Information Assurance Certification Paper

GIAC Intrusion Detection

Practical Assignment for SNAP San Jose

Submitted by: **Elizabeth Williams**
Date Submitted: **June 12, 2000**

TABLE OF CONTENTS

Detect 1 – Back Orifice

Jun 07 2000 13:13:36 : Deny inbound UDP from 212.211.0.27/1000 to x.x.x.2/31337
Jun 07 2000 13:13:36 : Deny inbound UDP from 212.211.0.27/1000 to x.x.x.31/31337
Jun 07 2000 13:13:36 : Deny inbound UDP from 212.211.0.27/1000 to x.x.x.31/31337

1. Source of trace:

http://www.sans.org/y2k/061000.htm (Daniel B. Holzman)

2. Detect was generated by:

I am unsure of this particular log format; however, the fields shown in the log are identified below.

Timestamp:	Jun 07 2000 13:13:36
Action Taken:	Packet denied inbound direction
Protocol:	UDP
Source IP Address:	212.211.0.27
Source Port:	1000
Destination IP Addresses:	x.x.x.2 and x.x.x.31
Destination Port:	31337

3. Probability the source address was spoofed:

This address is probably not spoofed. Within DNS, the name associated with the IP address most likely indicates a UUNET user connecting to a UUNET access server.

Name: mfs-pci-bqe-vty27.as.wcom.net
Address: 212.211.0.27

Since this is a Back Orifice attack, the hacker would want to know where he has been successful so that he can take control of the systems.

4. Description of attack:

This appears to be a Back Orifice attack. Back Orifice is a backdoor program developed by the "Cult of the Dead Cow" hacking group. It runs on hacker port 31337 by default, which spells out ELEET. (Although this is an improper spelling of elite, it is used to refer to elite hackers.) Back Orifice is a remote administration system that allows a user (hacker) to control a computer via a UDP connection.

With this attack, it appears that the hacker was scanning a subnet for any occurrences of Back Orifice. Note the rapid timestamp and the static source port of 1000.

5. Attack mechanism:

This attack is successful when a hacker finds a destination that responds to UDP port 31337.

Back Orifice infects systems in the manner of Trojan Horses. Back Orifice is small, and entirely self-installing. A person downloads or is sent the executable and the program invisibly runs. Simply clicking on the executable installs the server and hides it from the Windows "task list" and "close program list". When an attacker locates a server that is running Back Orifice, he can assume control.

Back Orifice 2000 is the latest release and is more flexible than the original Back Orifice. Back Orifice 2000 supports both TCP and UDP connections, offers more extensible plug-ins, and supports strong encryption. Further information on Back Orifice 2000 may be found at the following web site:

http://www.bo2k.com/indexwhatis.html

6. Correlations:

This particular detect was attributed to Daniel B. Holzman. He also reported a similar detect on June 9, 2000. The earlier trace may be seen at http://www.sans.org/y2k/060900.htm.

Various alerts describing Back Orifice are given below:

http://xforce.iss.net/alerts/advise31.php
http://www.cert.org/vul_notes/VN-98.07.backorifice.html
http://www.cert.org/summaries/CS-99-01.html, where UDP port 31337 is associated with Back Orifice.

Furthermore, Back Orifice was discussed in great detail at the Sans 2000 IDS Conference in San Jose, California.

7. Evidence of active targeting:

By looking at the snapshot of the log file, it appears that either the subnet is being targeted or two hosts on the subnet. It is most probable that the entire subnet is being targetted; however, there are not enough samples shown in the detect to state this conclusively.

8. Severity:

Severity is defined as **(Criticality + Lethality) – Countermeasures (System + Network)**. Each element within the formula is based on a 5-point scale.

For this detect, the severity is calculated as follows:

Element	Score	Remarks
Criticality	+3	The subnet, rather than specific machines, is being targeted.
Lethality	+5	The attacker can gain full control of the system if Back Orifice penetrates network defenses and is discovered running on systems.
System Countermeasures	-3	Systems probably have latest patches; however, system defense countermeasures are unknown.
Network Countermeasures	-5	Firewall prohibited the packets from entering into the protected network.
SCORE	0	

9. Defensive recommendation:

Since the packets were denied, it appears that the firewall rules protected the network from this attack.

10. Multiple choice test question:

If a hacker discovers Back Orifice 2000 running on one of your servers, what could he do to the server?

a.) Manage registry
b.) Log keystrokes
c.) Retrieve cached passwords from memory
d.) All of the above

Answer: d

Detect 2 – SUNRPC Port Probe

Jun 7 05:35:22 fwall 12 deny: TCP from 24.17.96.120.1788
to mysubnet.0.111 seq 8BABB3F2, ack 0x0, win 32120, SYN
Jun 7 05:35:22 fwall 15 deny: TCP from 24.17.96.120.1789
to mysubnet.1.111 seq 8BA6EB58, ack 0x0, win 32120, SYN
Jun 7 05:35:23 fwall 15 deny: TCP from 24.17.96.120.1790
to mysubnet.2.111 seq 8BC4971F, ack 0x0, win 32120, SYN
Jun 7 05:35:23 fwall 15 deny: TCP from 24.17.96.120.1791
to mysubnet.3.111 seq 8B9ED1AC, ack 0x0, win 32120, SYN
Jun 7 05:35:23 fwall 15 deny: TCP from 24.17.96.120.1792
to mysubnet.4.111 seq 8C290568, ack 0x0, win 32120, SYN

(… snip …)

Jun 7 05:35:35 fwall 15 deny: TCP from 24.17.96.120.2041
to mysubnet.253.111 seq 8B9AFE25, ack 0x0, win 32120, SYN
Jun 7 05:35:35 fwall 15 deny: TCP from 24.17.96.120.2042
to mysubnet.254.111 seq 8BED036B, ack 0x0, win 32120, SYN
Jun 7 05:35:36 fwall 12 deny: TCP from 24.17.96.120.1788
to mysubnet.0.111 seq 8BABB3F2, ack 0x0, win 32120, SYN

1. Source of trace:

http://www.sans.org/y2k/060900-1030.htm (Drew Brunson)

2. Detect was generated by:

I am unsure of this particular log format; however, the fields shown in the log are identified below.

Timestamp:	Jun 7 05:35:22
Host Name and FW Rule # Hit (Probably):	fwall 12
Action Taken:	Packet denied inbound direction
Protocol:	TCP
Source IP Address:	24.17.96.120
Source Port:	1788
Destination IP Addresses:	Mysubnet hosts 0-254
Destination Port:	111
Typical Sequence Number:	8BABB3F2
TCP Acknowledgement:	0x0 (No Acknowledgement)
TCP Window Size:	32120
TCP Flags:	SYN

3. Probability the source address was spoofed:

This address is probably not spoofed. Within DNS, the name associated with the IP address most likely indicates an @home user connecting to an @home access server in Omaha, Nebraska.

Name: cx478401-c.omhaw1.ne.home.com
Address: 24.17.96.120

Cable modem systems generally fall within the IP address range of 24.x.x.x. If the address was spoofed, the attacker would not know which servers on the subnet had the sunrpc service running. This attack relies on the successful completion of the TCP three-way handshake.

4. Description of attack:

In this instance, the attacker is looking to see which servers on a particular subnet are running the sunrpc service. It appears that the attacker has crafted packets and is using a script to send them to all servers on this subnet (and probably many others). The timestamp between packets is short and the hosts on the subnet being scanned are in sequential order, starting with host 0 and ending with host 254. Notice that the first hit to mysubnet.0 has the same sequence number as the next hit to mysubnet.0. Also, the source port on the first packet to mysubnet.0 is the same as the source port on the next packet to mysubnet.0. If sunrpc is discovered running on a server, then the attacker can use one of several known sunrpc exploits.

5. Attack mechanism:

RPC (Remote Procedure Call) was developed by Sun Microsystems and is used on most UNIX machines to build networked applications. Several programs use RPC (rpcbind and portmapper); therefore, there are ways that RPC can be exploited. The attacker is looking to see what UNIX systems are running RPC. If a system responds to the attacker, he would then probe deeper to see which RPC programs are running via a portmapper dump. For example, there is a rpc.cmsd overflow exploit. This exploit is an attempt to overflow a buffer on the Calendar Manager service.

6. Correlations:

This detect was attributed to Daniel B. Holzman. He also reported a similar detect on June 1, 2000, which may be viewed at http://www.sans.org/y2k/060100.htm.

Other alerts and advisories may be found at the web sites given below.

http://xforce.iss.net/alerts/vol-2_num-4.php#Sun-rpc.cmsd

http://www.cert.org/advisories/CA-99-08-cmsd.html

7. Evidence of active targeting:

This is a scan of an entire subnet looking for port 111 running. Chances are great that this is a script targeting many subnets, probably not these systems in particular.

8. Severity:

For this detect, the severity is calculated as follows:

Element	Score	Remarks
Criticality	+3	The subnet, rather than specific machines, is being targeted.
Lethality	+5	The attacker can exploit the system.
System Countermeasures	-3	Specific system defense countermeasures are unknown.
Network Countermeasures	-5	Firewall prohibited the packets from entering into the protected network.
SCORE	0	

9. Defensive recommendation:

The firewall sitting in front of the systems blocked external access to the RPC services. However, it would be advantageous to turn off all unnecessary RPC services.

10. Multiple choice test question:

On a UNIX system, how would you list all registered RPC services?

a) showmount -e
b) rpcinfo -p
c) grep sunrpc /etc/services
d) None of the above

Answer: b

Detect 3 – SYN-FIN Packet Scan

June 1 08:34:25.118877 210.196.222.18.53 > MyNet.3.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.140392 210.196.222.18.53 > MyNet.4.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.164549 210.196.222.18.53 > MyNet.5.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.174844 210.196.222.18.53 > MyNet.6.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.193856 210.196.222.18.53 > MyNet.7.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.218688 210.196.222.18.53 > MyNet.8.53:
SF 907639663:907639663(0) win 1028
June 1 08:34:25.242696 210.196.222.18.53 > MyNet.9.53:
SF 907639663:907639663(0) win 1028

1. Source of trace:

http://www.sans.org/y2k/060300.htm (Judith M. Ostroot)

2. Detect was generated by:

This detect came from a windump log. The fields shown or assumed from the log are identified below.

Timestamp:	June 1 08:34:25.118877
Action Taken:	Packet identified
Protocol:	TCP
Source IP Address:	210.196.222.18
Source Port:	53
Destination IP Addresses:	Hosts 3-9 on MyNet
Destination Port:	53
TCP Flags:	SYN-FIN
Typical Sequence Number:	907639663
Number of Bytes:	0
TCP Window Size:	1028

3. Probability the source address was spoofed:

This is a completely illegal combination of TCP flags. It is impossible for a normal implementation of TCP to generate packets like these. Therefore, it must be a crafted packet sent by an attacker (or attacker script) awaiting results and not spoofed.

According to DNS, the source IP address resolves as follows:

Name: dns1.udc-c.dion.ne.jp
Address: 210.196.222.18
Aliases: 18.222.196.210.in-addr.arpa

4. Description of attack:

In these scans the attacker uses an impossible flag combination to probe port 53 on a subnet. These packets must have been crafted. Upon inspection, the following is observed:

1. The sequence numbers and source port remains the same during the duration of the scan.
2. The destination hosts within the subnet are sequentially probed for destination port 53.
3. Timestamp of all hosts probed are virtually identical, which means that the attack was scripted.

This attack may occur for two reasons.

1. Some firewalls may be unable to detect the SF setting, which may enable the scan to penetrate further into the "protected" network.
2. The attacker may be hoping to generate a response from the illegal packet, thereby identifying the OS. For example, Linux machines respond to SYN-FIN with SYN-FIN-ACK.

5. Attack mechanism:

The attacker is sending crafted packets with SYN-FIN TCP flags set. If an attacker receives a response to these illegally crafted packets, then he may be able to determine the OS that sent the response. This is TCP/IP stack fingerprinting. Once the attacker discovers the OS, he is one step closer to compromising the system. He might do a port scan next on the targeted host to determine services that may be running.

6. Correlations:

This type of attack was covered by Vicki Irwin on day 2 of the SANS IDS Conference. Both QueSO and nmap may be used to send packets to the target and compare the responses with a database. Also, the links below provide additional information.

http://www.insecure.org/nmap/nmap-fingerprinting-article.html
http://www.securityfocus.com/templates/archive.pike?list=1&msg=Pine.LNX.3.96.980710165820.1382C-100000@think.kung.foo

7. Evidence of active targeting:

The attacker is targeting hosts on a subnet. He is probably attempting to ascertain the OS of targeted hosts. If he learns the OS, then he can choose vulnerabilities unique to the OS and exploit.

8. Severity:

For this detect, the severity is calculated as follows:

Element	Score	Remarks
Criticality	+3	Systems on targeted subnet are unknown.
Lethality	+3	If OS determined, specific exploits may be tried.
System Countermeasures	-1	No evidence that packets were denied. Systems may have responded.
Network Countermeasures	-1	No evidence that any firewall denied the packets.
SCORE	+4	

9. Defensive recommendation:

Defenses may be weak. Install network firewall and host based protection. Harden the OS as much as possible, install most recent security patches, and keep abreast of latest exploits.

10. Multiple choice test question:

According to RFC 793, what is the correct response for a server receiving a TCP packet with FIN set and ACK unset?

a) Send ACK
b) Send FIN-ACK
c) Do not respond
d) Send FIN

Answer: c

Detect 4 – Telnet Port Probe

1. Source of trace:

Jun 7 04:48:00 fwall 15 deny: TCP from 209.156.190.95.39557 to fwall.23
seq 105B2, ack 0x0, win 8192, SYN
Jun 7 04:48:09 fwall 15 deny: TCP from 209.156.190.95.39557 to fwall.23
seq 105B2, ack 0x0, win 8192, SYN
Jun 7 04:48:21 fwall 15 deny: TCP from 209.156.190.95.39557 to fwall.23
seq 105B2, ack 0x0, win 8192, SYN

2. Detect was generated by:

http://www.sans.org/y2k/060900-1030.htm (Drew Brunson)

I am unsure of this particular log format; however, the fields shown in the log are identified below.

Timestamp:	Jun 7 04:48:00
Host Name and FW Rule # Hit (Probably):	fwall 15
Action Taken:	Packet denied inbound direction
Protocol:	TCP
Source IP Address:	209.156.190.95
Source Port:	39557
Destination IP Addresses:	fwall
Destination Port:	23
Typical Sequence Number:	105B2
TCP Acknowledgement:	0x0 (No Acknowledgement)
TCP Window Size:	8192
TCP Flags:	SYN

3. Probability the source address was spoofed:

This address is probably not spoofed. According to Drew Brunson, this particular address came from his ISP. Within DNS, the name associated with the source IP address indicates a splitrock.net server.

Name: Woodlands95.splitrock.net
Address: 209.156.190.95

However, the system may be compromised. Instead of spoofing the address, the attacker may not be revealing his true location and using the woodlands95 server at splitrock to launch his attacks.

4. Description of attack:

With this scan the attacker is attempting to telnet into a target firewall that uses splitrock.net as an ISP. Note the following:

1. Timestamp of packets sent to the firewall are a few seconds apart. This is probably long enough to receive the "telnet connection refused" message.
2. The sequence number and source port remains the same during the duration of the attack.

The attacker may be checking to see if the firewall rules are misconfigured. A security engineer may enable telnet to the firewall from "inside the network" for administrative purposes.

5. Attack mechanism:

This is a telnet attempt. The attacker is attempting to initiate the three way handshake by sending a SYN packet to the firewall, requesting destination port 23, and using an ephemeral destination port of 39557. The attacker may be using this method for one of these reasons.

1. Gain access to the firewall, compromise it, further penetrate the interior defenses, etc. Once a system is compromised, anything is possible, e.g., DoS attack.
2. Determine the login banner for more targeted exploits.
3. Certain telnet exploits do not require a telnet login. For example, there is a buffer overflow attempt where the attacker inserts an incredibly long username. If successful, the attacker may crash the system or gain access to the system.

6. Correlations:

When I checked http://cve.mitre.org and searched the CVE and Candidate list for the telnet keyword, I came up with 21 matches. There are numerous ways to exploit telnet. The FAQ section on the SANS web site discusses the telnet pitfalls.

http://www.sans.org/newlook/resources/IDFAQ/telnet_rlogin.htm

Also, the reference below details a method of attacking a target running telnet.

http://rootshell.com/archive-j457nxiqi3gq59dv/199708/solaris_telnet.c.html

7. Evidence of active targeting:

Yes, there is clear evidence that the attacker is checking to see if the firewall is accessible via the telnet port.

8. Severity:

For this detect, the severity is calculated as follows:

Element	Score	Remarks
Criticality	+5	Firewall was probed.
Lethality	+5	Attacker could do serious damage if firewall was compromised or if it suffered DoS attack.
System Countermeasures	-3	Unknown if all patches applied, type of firewall OS, etc.
Network Countermeasures	-5	Firewall denied telnet requests.
SCORE	+2	

9. Defensive recommendation:

The firewall denied the telnet attempt. Additional security measures that may apply include:

1. Hardening the firewall OS.
2. Replacing telnet with SSH.
3. Using TCP wrappers on the firewall.
4. Installing latest security patches.

10. Multiple choice test question:

Which of the following is true about telnet?

a.) Less secure than SSH
b.) Used to make connections between machines on a network
c.) Uses TCP port 23
d.) All of the above

Answer: d

Detect 5 – Akamai Proximity Detection

```
May 25 19:36:41 icmp 206.128.186.172 -> 203.23.109.34 (8/0), 1 packet
May 25 19:36:46 icmp 216.32.119.15 -> 203.23.109.34 (8/0), 1 packet
May 25 19:37:04 icmp 206.191.161.41 -> 203.23.109.34 (8/0), 1 packet
May 25 19:37:15 icmp 212.133.25.139 -> 203.23.109.34 (8/0), 1 packet
May 25 19:37:37 icmp 63.211.120.42 -> 203.23.109.34 (8/0), 1 packet
May 25 19:37:51 icmp 216.32.16.15 -> 203.23.109.34 (8/0), 1 packet
May 25 19:37:52 icmp 204.201.228.130 -> 203.23.109.34 (8/0), 1 packet
May 25 19:38:03 icmp 208.178.144.138 -> 203.23.109.34 (8/0), 1 packet
May 25 19:38:14 icmp 216.200.14.139 -> 203.23.109.34 (8/0), 1 packet
May 25 19:38:27 icmp 216.52.232.137 -> 203.23.109.34 (8/0), 1 packet
May 25 19:38:53 icmp 206.132.160.42 -> 203.23.109.34 (8/0), 1 packet
May 25 19:38:55 icmp 216.32.65.143 -> 203.23.109.34 (8/0), 1 packet
May 25 19:39:14 icmp 195.22.196.105 -> 203.23.109.34 (8/0), 1 packet
```

1. Source of trace:

http://www.sans.org/y2k/052700-2100.htm (Phil Crooker)

2. Detect was generated by:

I am unsure of this particular log format; however, the fields shown in the log are identified below.

Timestamp:	May 25 19:36:41
Protocol:	ICMP
Source IP Address:	206.128.186.172 (and others)
Destination IP Addresses:	203.23.109.34
Number of Packets per log entry:	1 packet

3. Probability the source address was spoofed:

No, the source addresses were not spoofed. I believe that it is an attempt to determine latency between the various Akamai servers and the destination IP address.

4. Description of attack:

I do not believe that this is an attack. Sampling the source IP addresses reveals that they are affiliated with Akamai (http://www.akamai.com). Akamai's business involves deploying caching servers throughout the world and bringing web content closer to the users. I believe that Akamai also announced the ability for their customers to view the geographic origin of web site visitors.

Performing an nslookup on the destination IP address revealed the following:

Name: mail.orix.com.au
Address: 203.23.109.34

Here is the DNS resolution for some of the source IP addresses.

Name: a216-32-119-15.deploy.akamaitechnologies.com
Address: 216.32.119.15

Name: a206-191-161-41.deploy.akamaitechnologies.com
Address: 206.191.161.41

Name: a63.211.120.42.deploy.akamaitechnologies.com
Address: 63.211.120.42

Name: a216-32-16-15.deploy.akamaitechnologies.com
Address: 216.32.16.15

Name: a204-201-228-130.deploy.akamaitechnologies.com
Address: 204.201.228.130
Aliases: 130.228.201.204.in-addr.arpa

5. Attack mechanism:

This appears to be an attempt at determining latency between the various Akamai source servers and destination 203.23.109.34. While some would question the accuracy of using ping to determine latency, it is still used. It seems that Akamai is sending a single icmp packet from multiple source addresses to destination 203.23.109.34. Akamai probably has an algorithm that correlates the ping data and determines the caching server that is closest to the destination IP address.

6. Correlations:

There are a number of geographic load balancers and caching servers that utilize a latency measurement method to determine the closest server to the end user.

For example, here is a link to the 3DNS load balancer product by F5.

http://www.f5.com/3dns/index.html

In addition, Digital Island has a product called Footprint that utilizes caching technology to deploy content at the edges, e.g., dial-up ISPs.

Here is a link to Footprint by Digital Island.

http://www.digisle.net/services/cd/footprint.shtml

7. Evidence of active targeting:

This is not malicious. It is an attempt to determine latency between the destination IP address and the various Akamai servers.

8. Severity:

For this detect, the severity is calculated as follows:

Element	Score	Remarks
Criticality	+3	Log showed ICMP packets to a mail server.
Lethality	+1	ICMP is used in DoS attacks; this is latency measurement.
System Countermeasures	-2	OS of destination IP address is unknown. Security patches may not be applied.
Network Countermeasures	-2	ICMP entries appeared in log. Firewall is letting ICMP packets arrive at a mail server.
SCORE	0	

9. Defensive recommendation:

ICMP packets were logged and no harm took place. In general, one should be very selective about which servers are allowed to receive ICMP packets given the recent Denial of Service attacks. Also set up router access control lists, such that ICMP is either blocked or restricted.

10. Multiple choice test question:

Which of the following is true?

a) Load balancing is synonymous with caching
b) Every log entry is indicative of an attack
c) Load balancers, caching, mirroring, and content delivery are tools used to enhance the end user web experience
d) None of the above

Answer: c

Detect 6 – Proxy Probe/Ring Zero

Site: @home Host lookup: Date: 20000330 Pattern:
src host 63.11.117.219 /usr/local/logger/one_day_pat.pl
-S -d 20000330 -l @home -p 'src host 63.11.117.219 '
/Shadow/@home/Mar30
16:55:35.440651 1Cust219.tnt1.bryan.oh.da.uu.net.1865 > @.home.com.8080:
S 12492586:12492586(0) win 8192 (DF)
16:55:35.465692 1Cust219.tnt1.bryan.oh.da.uu.net.1866 > @.home.com.www:
S 12492589:12492589(0) win 8192 (DF)
16:55:35.484070 1Cust219.tnt1.bryan.oh.da.uu.net.1868 > @.home.com.3128:
S 12492595:12492595(0) win 8192 (DF)
16:55:35.484222 1Cust219.tnt1.bryan.oh.da.uu.net.1870 > @.home.com.8050:
S 12492601:12492601(0) win 8192 (DF)
16:55:35.484367 1Cust219.tnt1.bryan.oh.da.uu.net.1869 > @.home.com.8002:
S 12492598:12492598(0) win 8192 (DF)
16:55:36.664311 1Cust219.tnt1.bryan.oh.da.uu.net.1865 > @.home.com.8080:
S 12492586:12492586(0) win 8192 (DF)
16:55:36.666792 1Cust219.tnt1.bryan.oh.da.uu.net.1868 > @.home.com.3128:
S 12492595:12492595(0) win 8192 (DF)
16:55:36.706460 1Cust219.tnt1.bryan.oh.da.uu.net.1869 > @.home.com.8002:
S 12492598:12492598(0) win 8192 (DF)
16:55:36.758762 1Cust219.tnt1.bryan.oh.da.uu.net.1870 > @.home.com.8050:
S 12492601:12492601(0) win 8192 (DF)
16:55:37.625224 1Cust219.tnt1.bryan.oh.da.uu.net.1865 > @.home.com.8080:
S 12492586:12492586(0) win 8192 (DF)
16:55:37.939332 1Cust219.tnt1.bryan.oh.da.uu.net.1868 > @.home.com.3128:
S 12492595:12492595(0) win 8192 (DF)
16:55:37.949391 1Cust219.tnt1.bryan.oh.da.uu.net.1869 > @.home.com.8002:
S 12492598:12492598(0) win 8192 (DF)
16:55:37.985345 1Cust219.tnt1.bryan.oh.da.uu.net.1870 > @.home.com.8050:
S 12492601:12492601(0) win 8192 (DF)
16:55:38.396403 1Cust219.tnt1.bryan.oh.da.uu.net.1866 > @.home.com.www:
S 12492589:12492589(0) win 8192 (DF)
16:55:38.786750 1Cust219.tnt1.bryan.oh.da.uu.net.1865 > @.home.com.8080:
S 12492586:12492586(0) win 8192 (DF)

1. Source of trace:

http://www.sans.org/y2k/033100.htm (Guy Bruneau)

2. Detect was generated by:

This detect came from a system running Shadow. In particular, the analyst ran the one-day pattern perl script for March 30, 2000. Site is @home and pattern specified is Source IP address 63.11.117.219.

Timestamp:	16:55:35.440651
Protocol Implied:	TCP
Source Host:	1Cust219.tnt1.bryan.oh.da.uu.net
Source Ports:	Port range 1865-1870

Destination:	@home.com
Destination Port:	8080, 80, 3128, 8050, 8002
TCP Flags:	SYN
Typical Sequence Number:	12492586
Number of Bytes	0
TCP Window Size	8192
IP Fragment Flag	DF (Don't Fragment)

3. Probability the source address was spoofed:

No, the source IP address was not spoofed.

4. Description of attack:

The attacker is sending out SYN packets. Importing log data into Excel enables further analysis. For example, the table below has been sorted by destination port.

16:55:35.484	1Cust219.tnt1.bryan.oh.da.uu.net.1868	>	@.home.com.3128:	S	12492595:12492595(0)
16:55:36.667	1Cust219.tnt1.bryan.oh.da.uu.net.1868	>	@.home.com.3128:	S	12492595:12492595(0)
16:55:37.939	1Cust219.tnt1.bryan.oh.da.uu.net.1868	>	@.home.com.3128:	S	12492595:12492595(0)
16:55:35.484	1Cust219.tnt1.bryan.oh.da.uu.net.1869	>	@.home.com.8002:	S	12492598:12492598(0)
16:55:36.706	1Cust219.tnt1.bryan.oh.da.uu.net.1869	>	@.home.com.8002:	S	12492598:12492598(0)
16:55:37.949	1Cust219.tnt1.bryan.oh.da.uu.net.1869	>	@.home.com.8002:	S	12492598:12492598(0)
16:55:35.484	1Cust219.tnt1.bryan.oh.da.uu.net.1870	>	@.home.com.8050:	S	12492601:12492601(0)
16:55:36.759	1Cust219.tnt1.bryan.oh.da.uu.net.1870	>	@.home.com.8050:	S	12492601:12492601(0)
16:55:37.985	1Cust219.tnt1.bryan.oh.da.uu.net.1870	>	@.home.com.8050:	S	12492601:12492601(0)
16:55:35.441	1Cust219.tnt1.bryan.oh.da.uu.net.1865	>	@.home.com.8080:	S	12492586:12492586(0)
16:55:36.664	1Cust219.tnt1.bryan.oh.da.uu.net.1865	>	@.home.com.8080:	S	12492586:12492586(0)
16:55:37.625	1Cust219.tnt1.bryan.oh.da.uu.net.1865	>	@.home.com.8080:	S	12492586:12492586(0)
16:55:38.787	1Cust219.tnt1.bryan.oh.da.uu.net.1865	>	@.home.com.8080:	S	12492586:12492586(0)
16:55:35.466	1Cust219.tnt1.bryan.oh.da.uu.net.1866	>	@.home.com.www:	S	12492589:12492589(0)
16:55:38.396	1Cust219.tnt1.bryan.oh.da.uu.net.1866	>	@.home.com.www:	S	12492589:12492589(0)

The following may be ascertained from the log entries.

1. Timestamps between packets are extremely close together.
2. Source IP address remains the same throughout the detect.
3. Source port 1865 is always associated with destination port 8080, using the same sequence number each time for this port combination.
4. Source port 1866 is always associated with destination port 80, using the same sequence number each time for this port combination.
5. Source port 1868 is always associated with destination port 3128, using the same sequence number each time for this port combination.
6. Source port 1869 is always associated with destination port 8002, using the same sequence number each time for this port combination.
7. Source port 1870 is always associated with destination port 8050, using the same sequence number each time for this port combination.

This could be a Ring Zero trojan probe looking for active proxy servers on port 80, 8080, and 3128. Another possibility is that this is a web user being served by a squid web proxy cache.

5. Attack mechanism:

If this is a Ring Zero attack, there would most likely be code/files left on a server with the following names: ITS.EXE and RING0.VXD. The code ITS.EXE attempts to remove files from a webserver. There is also a file called PST.EXE used to discover proxies and send the proxy IP addresses to another location for future use.

6. Correlations:

Additional information on Ring Zero may be obtained from the following web sites:

http://unhinfo.unh.edu/cis-workstation/security/ringzero.html
http://server1.sans.org/newlook/resources/IDFAQ/ring_zero.htm

Also, http://www.simovits.com maintains a list of trojan horses detailing the following:

Port 80	Back End, Executor, Hooker, RingZero
Port 3128	RingZero
Port 8080	RingZero

7. Evidence of active targeting:

Given the proliferation of cache servers in various ISPs, this may be a detect showing a web user served from squid cache. However, without knowing specific destination IP addresses targeted, this could be Ring Zero.

8. Severity:

Assuming this is Ring Zero, the severity is calculated as follows:

Element	Score	Remarks
Criticality	+3	Targeting web servers.
Lethality	+4	Ring Zero could certainly turn a web server into a sacrificial lamb.
System Countermeasures	-2	OS of destinations unknown. Security patches may not be applied.
Network Countermeasures	-3	These scans are logged and monitored by an IDS. Network defenses are somewhat unknown.
SCORE	+2	

9. Defensive recommendation:

Block inbound packets on port 8080 and 3128 at the firewall if the site does not use proxies. If proxy traffic does need to pass through the internal network, restrict it with firewall rules.

10. Multiple choice test question:

Which of the following is not true?

a) Squid proxy cache is usually associated with TCP port 3128
b) A proxy web server is usually associated with TCP port 8080
c) A web server is usually associated with TCP port 8080
d) Intrusion analysts may disagree on the interpretation of a detect

Answer: c

Detect 7 – Linuxconf Scan

```
May 31 00:20:44 zion snort[27540]: spp_portscan:
  PORTSCAN DETECTED from 210.112.192.74
May 31 00:20:44 210.112.192.74:1773 -> x.y.z.100:98 SYN **S*****
May 31 00:20:44 210.112.192.74:1778 -> x.y.z.105:98 SYN **S*****
May 31 00:20:44 210.112.192.74:1780 -> x.y.z.107:98 SYN **S*****
May 31 00:20:44 210.112.192.74:1775 -> x.y.z.102:98 SYN **S*****
May 31 00:20:44 210.112.192.74:1793 -> x.y.z.120:98 SYN **S*****
May 31 00:22:39 zion snort[27540]: spp_portscan: portscan status
  from 210.112.192.74: 9 connections across 9 hosts: TCP(9), UDP(0)
May 31 00:22:44 zion snort[27540]: spp_portscan: End of portscan
  from 210.112.192.74
May 31 11:41:45 zion snort[27540]: spp_portscan:
  PORTSCAN DETECTED from 210.112.192.74
```

1. Source of trace:

http://www.sans.org/y2k/060100-1400.htm (Sean Brown)

2. Detect was generated by:

This detect came from a system running snort. Snort indicated a portscan was detected from 210.112.192.74. Fields in the log indicate the following:

Timestamp:	May 31 00:20:44
Protocol Implied:	TCP
Source IP Address:	210.112.192.74
Source Ports:	Port range 1700-1799
Destination:	x.y.z.100 (9 unique hosts on x.y.z network)
Destination Port:	98
TCP Flags:	SYN

3. Probability the source address was spoofed:

No, the source IP address was not spoofed. The attacker would need to know which of his targeted hosts responded to the TCP port 98 scan. Using traceroute, the last registered router shows the path through lginternet.channeli.com. This domain uses the Public DNS Service at http://www.granitecanyon.com. Setting the DNS server to ns1.granitecanyon.com and looking for 210.112.192.74 yielded no results. However, Korea was the last known country that could be determined using traceroute and nslookup.

4. Description of attack:

This is a Linuxconf scan from a single source IP address to nine different hosts on the same subnet. It also appears that another Linuxconf scan was initiated from the same source IP address several hours later. Note the timestamps showing two Linuxconf scans.

Linuxconf is an administration tool for the Linux operating system. The creators of Linuxconf consider it both a configurator and an activator. It determines when daemons should be started, killed, etc., as well as executes many other common configuration tasks. For example, it can be used to reconfigure a network interface, remount a volume, and remove a network route.

5. Attack mechanism:

The attacker is scanning the systems on the subnet to see if the Linux remote configuration service is available, as evidenced by the connection attempts to TCP port 98. Linuxconf is used to manage remote systems and runs as root. If an attacker can gain access to the Linuxconf utility on a system, he can compromise it.

6. Correlations:

Linuxconf and its vulnerabilities are widely known. Further information may be found at http://lwn.net/1999/1223/a/linuxconfresponse.html. It has also been discussed on bugtraq.

Linuxconf capabilities are discussed in greater detail at http://www.solucorp.qc.ca/linuxconf/.

7. Evidence of active targeting:

Yes, this is targeting. The attacker wants to determine which hosts on a subnet will respond to the Linuxconf port scan. We don't know if the attacker has already done OS fingerprinting and is only targeting Linux servers.

8. Severity:

The severity is calculated as follows:

Element	Score	Remarks
Criticality	+4	Linuxconf scan.
Lethality	+4	Linux hosts could be compromised.
System Countermeasures	-3	IDS is logging the scans. Don't know if all system countermeasures (patches, etc.,) have been applied.
Network Countermeasures	-4	Assumption is IDS positioned outside the firewall and that linuxconf portscan cannot penetrate into interior network.
SCORE	+1	

9. Defensive recommendation:

Block linuxconf port probes at the firewall. If linuxconf remote administration is a must, encrypt the traffic. Set up intrusion detection in front and behind the firewall. Stay abreast of latest exploits, including the candidates at http://cve.mitre.org. Finally, one could disable linuxconf network access using its native utility.

10. Multiple choice test question:

Which of the following are scanning tools used to gather information on targeted systems?

a) Nmap
b) QueSO
c) Strobe
d) All of the above

Answer: d

Detect 8 – Smurf Amplification

06/08/2000 15:23:23.624 - Smurf Amplification Attack Dropped -
Source:151.4.157.1, 8, WAN - Destination:255.255.255.255, 8, LAN - -

1. Source of trace:

http://www.sans.org/y2k/061100.htm (Bill Stewart)

2. Detect was generated by:

This detect came from a firewall log. I am unsure of which firewall; however, key fields are detailed below.

Timestamp:	06/08/2000 15:23:23.624
Information:	Smurf Amplification Attack Dropped
Source IP Address:	151.4.157.1
Destination IP Address:	255.255.255.255

3. Probability the source address was spoofed:

Yes, the source address was most likely spoofed. When a router receives a packet, it looks at the destination IP address to know where to forward the packet. The source address is used by the destination in responding back. With a smurf attack, the attacker does not want to see the responses.

4. Description of attack:

As evidenced by the log entry, a Smurf Amplication attack was attempted. Smurf is based on IP spoofing and broadcasts. An attacker sends a single packet to a broadcast address. All machines on the LAN respond to the broadcast. By spoofing the source IP address, responses are not returned to the attacker. Instead they are returned to the source IP address.

5. Attack mechanism:

The Smurf attack is a denial of service that impacts a target network and an intermediary network. The attacker spoofs an IP address and floods the intermediary network with broadcast echo requests. If the attack penetrates the firewall or screening router, many of the hosts on the subnet will respond back to the real holder of the source IP address. This enables the attacker to retain his anonymity and continue attacking other networks. Some of the effects of this targeted attack are detailed below.

1. Hosts on the targeted network are bogged down answering the abundance of requests they are receiving.
2. Network links become saturated with “no value” packets.
3. The machine who legitimately “owns” the source IP address is receiving all the replies.
4. Secondarily, ICMP packets received by the intermediary network may be malformed thereby causing some systems to crash.

6. Correlations:

This is extremely well known. The heading for the attack in the SANS 2.4/2.5 IDS San Jose Conference book is, “Boring Ol Smurf”.

The following URLs discuss Smurf and validate findings.

http://www.cisco.com/warp/public/707/22.html
http://www.cisco.com/warp/public/707/newsflash.html
http://www.cert.org/advisories/CA-98.01.smurf.html
http://www.codetalker.com/whitepapers/dos-smurf.html
http://users.quadrunner.com/chuegen/smurf.cgi
http://www.nfr.net/firewall-wizards/mail-archive/1999/Feb/0086.html

7. Evidence of active targeting:

Yes, this is active targeting. The Smurf attack is a Denial of Service that affects the amplifier (intermediary) network and the real “owner” of the spoofed IP address. Most likely, there are multiple spoofed IP addresses and this problem is compounded.

8. Severity:

The severity is calculated as follows:

Element	Score	Remarks
Criticality	+5	Smurf attack.
Lethality	+5	Denial of service for intermediary network, target network, and potentially intervening networks.
System Countermeasures	-5	Probably good. Notification was sent to GIAC, which means log files are monitored.
Network Countermeasures	-5	Smurf amplication attack did not succeed.
SCORE	0	

9. Defensive recommendation:

Set up ICMP echo filters in the screening router or firewall and filter incoming broadcasts. On Cisco routers, specify “no ip directed-broadcast” (interface command) and explicitly deny traffic destined for broadcast addresses behind the filtering router. The interface command “no ip directed-broadcast” prevents a router from converting the broadcast echo requests to hardware level broadcasts.

Equally important, configure the external router to block all outbound packets from your site that indicate a source address not within your IP address space.

Also, check http://www.netscan.org to make sure your site is not listed as a Smurf amplifier site.

10. Multiple choice test question:

If one could categorize attacks by protocol, where would the Smurf attack fit?

a) TCP
b) ICMP
c) UDP
d) IP

Answer: b

Detect 9 – IRC/Denial of Service

02:18:30.295451 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:30.301084 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:30.306508 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:30.312112 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:30.317681 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:30.323070 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024

[Snip more packets]

02:18:31.100991 morannon.kdi.com > 209.216.2.200: icmp: morannon.kdi.com udp port ircd unreachable [tos 0xc0]
02:18:31.106472 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:31.111972 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024
02:18:31.117526 209.216.2.200.4164 > morannon.kdi.com.ircd: udp 1024

02:26:31.574840 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30039:1480@0+)
02:26:31.574847 209.216.2.200 > morannon.kdi.com:
(frag 30041:48@2960)
02:26:31.583572 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30041:1480@0+)
02:26:31.583582 209.216.2.200 > morannon.kdi.com:
(frag 30044:48@2960)
02:26:31.591760 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30044:1480@0+)
02:26:31.591768 209.216.2.200 > morannon.kdi.com:
(frag 30046:48@2960)
02:26:31.600166 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30046:1480@0+)
02:26:31.600173 209.216.2.200 > morannon.kdi.com:
(frag 30048:48@2960)
02:26:31.609754 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30048:1480@0+)
02:26:31.609785 209.216.2.200 > morannon.kdi.com:
(frag 30050:48@2960)
02:26:31.618328 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30050:1480@0+)
02:26:31.618354 209.216.2.200 > morannon.kdi.com:
(frag 30052:48@2960)
02:26:31.626650 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30052:1480@0+)
02:26:31.626656 209.216.2.200 > morannon.kdi.com:
(frag 30054:48@2960)
02:26:31.635248 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30054:1480@0+)
02:26:31.635253 209.216.2.200 > morannon.kdi.com:
(frag 30056:48@2960)
02:26:31.643568 209.216.2.200 > morannon.kdi.com:
icmp: echo request (frag 30056:1480@0+)
02:26:31.643574 209.216.2.200 > morannon.kdi.com:
(frag 30058:48@2960)
02:26:31.652679 209.216.2.200 > morannon.kdi.com:

icmp: echo request (frag 30058:1480@0+)
02:26:31.652687 209.216.2.200 > morannon.kdi.com:
(frag 30060:48@2960)

1. Source of trace:

http://www.sans.org/y2k/032900.htm

2. Detect was generated by:

This detect came from tcpdump. Key fields for the ircd packets are shown below.

Timestamp:	02:18:30.295451
Source IP Address:	209.216.2.200
Source Port:	4164
Destination IP Address:	morannon.kdi.com
Destination Port:	IRCD (6667)
Protocol:	UDP

3. Probability the source address was spoofed:

Yes, there is a chance that the IP address was spoofed or bounced. There were thousands of packets sent to UDP port 6667 within a 3-minute timeframe. Several minutes later, the IRCD packets were followed by thousands of fragmented ICMP requests within a 4-minute timeframe.

4. Description of attack:

It appears that source host 209.216.2.200 is looking for UDP port 6667 (IRCD) at morannon.kdi.com. Several minutes later, the same source is sending thousands of fragmented ICMP requests to morannon.kdi.com.

5. Attack mechanism:

IRC stands for Internet Relay Chat. It is a multi-user, multi-channel chat system (ran over a network) that gives people all over the world the ability to talk (type online) to one another in real time. Each user has a username and talks with other users privately or on a channel (chat room). Hackers know that all channels are not registered and subject to takeovers. To show dominance, hackers may try to kick everyone off the channel and essentially take it over. In particular, hackers may flood the IRC channel or attempt a DoS attack.

Because IRC exposes IP addresses to others in chatrooms, some may target DoS attacks against these addresses. For example, the Ping of Death DoS attack consists of sending fragments that are impossible to reassemble. With Ping of Death, the sum total of the fragments for a single packet exceeds the maximum IP packet size of 65535 bytes. If this were to occur on a target host, the buffer would overflow causing the system to crash, freeze, or reboot. A fragmented packet may also bypass some firewalls acting as "packet filters".

6. Correlations:

A good description of how IRC can be used is http://www.stanford.edu/~dbrumley/Me/irc.txt.

I didn't locate a similar detect on the SANS web site. However, additional information on IRC may be obtained from the following web sites.

http://209.143.242.119/cgi-bin/search/search.cgi?searchvalue=irc&type=archives
http://xforce.iss.net/static/4320.php
http://packetstorm.securify.com/irc/

7. Evidence of active targeting:

Yes, this is host targeting. Log indicates that attacker was first looking for IRCD. He then sent thousands of fragmented ICMP packets.

8. Severity:

The severity is calculated as follows:

Element	Score	Remarks
Criticality	+3	Host system unknown.
Lethality	+5	Denial of service.
System Countermeasures	-5	Probably good. Notification was sent to GIAC, which means log files are monitored.
Network Countermeasures	-5	Assumption is that the IDS was located outside the screening router/firewall and caught these attempts. Further assumption is that firewall would drop these packets.
SCORE	-2	

9. Defensive recommendation:

Block port 6667 at the screening firewall, as well as install ICMP filters. Also, create an IDS filter to monitor port 6667.

10. Multiple choice test question:

What is a script kiddie?

a) Hacker wannabe
b) Would be malicious programmer
c) Someone just above the skill level of trained monkeys
d) All of the above

Answer: d

Detect 10 – DNS Zone Transfer

May 22 02:11:24 [fw] May 22 2000 02:12:26: %PIX-2-106001:
Inbound TCP connection denied from 195.163.20.184/65535
to serve1/53 flags SYN
May 22 02:11:25 [fw] May 22 2000 02:12:28: %PIX-2-106001:
Inbound TCP connection denied from 195.163.20.184/65535
to serve2/53 flags SYN
May 22 02:11:25 [fw] May 22 2000 02:12:28: %PIX-7-106011:
Deny inbound (No xlate) tcp src outside:195.163.20.184/65535
dst outside:global/53

1. Source of trace:

http://www.sans.org/y2k/052700-2100.htm (Roger Lutz)

2. Detect was generated by:

This detect came from what appears to be a PIX firewall log. Key fields are detailed below:

Timestamp:	May 22 02:11:24
Information:	Inbound TCP connection denied
Source IP Address:	195.163.20.184
Source Port:	65535
Destination IP Address:	serve1,serve2
Destination Port:	53
TCP Flags:	SYN

3. Probability the source address was spoofed:

It's highly unlikely the IP address was spoofed. The DNS response would need to go back to the attacker. DNS shows source IP address resolution as follows:

Name: pc184.net20.ktv.koping.se
Address: 195.163.20.184

4. Description of attack:

Examining the log reveals an attempt by a Kabel modem user in Sweden to connect to TCP port 53 on both serve1 and serve2 and initiate a DNS zone transfer. With the advent of BINDv8, the local nameserver selects a random port above 1023 to initiate a connection to another nameserver on port 53.

TCP is used when a remote name server detects that its response to a server to server query would be greater than 512 bytes. When this occurs, the remote name server sends back a UDP packet telling the local name server to retry the query using TCP.

5. Attack mechanism:

A Kabel modem user in Sweden initiated these DNS zone transfer requests as part of the three-way TCP handshake. Pointing the web browser to http://www.ktv.koping.se shows that Koping is a Kabel TV/Internet provider located in Sweden. This user is probably pretending to be a DNS server hoping to discover the DNS records of the targeted domain. In this instance, perhaps the attacker already did a reconnaissance DNS port scan and learned about these particular DNS servers. By initiating a DNS zone transfer, the attacker could further map out the network.

The zone transfer download would enable the attacker to know about the machines listed in the DNS server. This list may include those machines within the site-specific domain, as well as machines that are outside the site-specific domain.

6. Correlations:

Here are a couple DNS postings submitted to SANS.

http://www.sans.org/y2k/053000-1000.htm, where L. Christopher Paul submitted zone transfer scan.
http://www.sans.org/y2k/052100.htm where Lisa Yeo submitted log file entries with source port 65535.

7. Evidence of active targeting:

This appears to be directed toward at least two DNS servers and is not a DNS port scan.

8. Severity:

The severity is calculated as follows:

Element	Score	Remarks
Criticality	+5	DNS Servers.
Lethality	+4	Zone transfer would provide a targeted list to begin attacking.
System Countermeasures	-4	Assumed to be good. Don't know how nameserver is configured, so there may be room for improvement.
Network Countermeasures	-5	Firewall dropped the DNS zone transfer requests.
SCORE	0	

9. Defensive recommendation:

Configure DNS nameservers to limit zone transfers to known hosts. Don't force external nameservers to retry with TCP. Also, consider implementing a "split-DNS" strategy, separating internal DNS servers from external ones. If an external DNS server is compromised, there is still another layer of protection.

10. Multiple choice test question:

A remote nameserver tells a local name server to retry the query using TCP when it detects that the UDP packet would be larger than:

a) 64 bytes
b) 128 bytes
c) 256 bytes
d) 512 bytes

Answer: d