Intrusion Detection Practical Assignment

For SANS Security DC2000

Dustin C. Childs

Introduction

This paper qualifies as the practical assignment for the intrusion detection track of the SANS Security DC2000 conference. This assignment has been divided into three parts. The first part shows 5 network traces from my local network. The first two of these traces show exploits that are listed in the SANS top ten. The second section of this assignment shows an analysis of an exploit. The last section shows an analysis of Snort logs provided by SANS from http://www.sans.org/giactc/snort/index.htm.

NOTE: All host names for network traces, both source and destination, have been sanitized. For the purposes of this exercise, the initiating source will always be a workstation in "badguy.com". All of the targets are listed as being in the "goodguy.com" domain. The IP address are shown as IANA Reserved addresses. The source IP addresses are shown as beginning with 10.0.0. All destination addresses are shown as beginning with 192.168.

Section One - Network Traces

Trace One

```
17:07:13.224109 P stalin.badguy.com.1027 > protect-
5.goodguy.com.domain: 41036 inv_q+ [b2&3=0x980] A? . (27)
                   4500 0037 22ea 0000 4011 6685 0a00 009a
                   c0a8 2605 0403 0035 0023 ecda a04c 0980
                   0000 0001 0000 0000 0000 0100 0100 007a
                   6900 0404 0302 01
17:07:13.227293 P protect-5.goodguy.com.domain >
stalin.badguy.com.1027: 41036 inv_q Refused [0q] 1/0/0 (27) (DF)
                   4500 0037 1f79 4000 fd11 6cf5 c0a8 2605
                   0a00 009a 0035 0403 0023 6cd5 a04c 8985
                   0000 0001 0000 0000 0000 0100 0100 007a
                   6900 0404 0302 01
17:07:13.227578 P stalin.badguy.com.1027 > protect-
5.goodguy.com.domain: 1974+ [b2&3=0x180] TXT CHAOS)? version.bind. (30)
                   4500 003a 22eb 0000 4011 6681 0a00 009a
                   c0a8 2605 0403 0035 0026 df47 07b6 0180
                   0001 0000 0000 0000 0776 6572 7369 6f6e
                   0462 696e 6400 0010 0003
17:07:13.232154 P protect-5.goodguy.com.domain >
stalin.badguy.com.1027: 1974* 1/0/0 CHAOS) TXT BIND 8.1.2 (65) (DF)
```

```
4500 005d 1f7a 4000 fd11 6cce c0a8 2605
0a00 009a 0035 0403 0049 a704 07b6 8580
0001 0001 0000 0000 0776 6572 7369 6f6e
0462 696e 6400 0010 0003 0756 4552 5349
4f4e 0442 494e 4400 0010 0003 0000 0000
000b 0a42 494e 4420 382e 312e 32
```

1. Source of the trace:

My Network

2. Detect

This detect was generated by Snort version 1.6 using the base rule set from http://www.snort.org.

3. Probability the source address was spoofed:

It is unlikely the source address was spoofed because the attacker is expecting information back from the target. The address may have been spoofed, but the attacker would not retrieve any information from the intended target.

4. Description of attack:

This trace shows an attacker doing some reconnaissance on a DNS server. The attacker is looking for the specific version of Bind running on the DNS server. First, an inverse query is sent to see if the server will accept this type of queries. The server returns a "inv_q Refused" message saying it will not. Next, the attacker attempts to find the version of named running. In the case shown above, the server responds with "TXT BIND 8.1.2" – showing the attacker the server is indeed running BIND version 8.1.2. An attacker can now only run exploits that target this version of bind. Based on the packet signature, we can guess that the tool "binfo-udp.c" (http://packetstorm.securify.com/Exploit_Code_Archive/binfo-udp.c) or some derivative was used.

5. Attack Mechanism

A buffer overflow exists in certain versions of BIND – specifically, versions previous to 8.2. BIND fails to properly bound the data received when processing an inverse query. Upon a memory copy, portions of the program can be overwritten, and arbitrary commands run on the affected host. Because of this, many attackers attempt to find the BIND version number before running an actual exploit. This is seen both in the inverse query and the query to version.bind. Interestingly, Adam McKenna has released a patch for BIND 8.1.2 (shown above) that will return "Go Away" when queried in this manner and write an "attackalert" message in the DNS log with the IP of the perpetrator (http://www.securityfocus.com/templates/archive.pike?list=1&date=1998-06-8&thread=Pine.LNX.3.96.980610171407.8195A-100000@dolemite.psionic.com).

6. Correlations

The inverse query buffer overflow exploit is described in CVE 1999-009 (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0009). This source IP has also been seen to OS Fingerprinting on the rest of the Class C.

7. Evidence of Active Targeting

There is a good deal of evidence for active targeting here. The attacker directed this attack specifically at a DNS server running BIND, not just any random machine. They must have already gathered enough information about the network to determine this machine was the DNS server.

8. Severity

Severity = (Criticality + Lethality) - (System Countermeasures + Network Countermeasures)
(5 + 2) – (2 + 2) = 3

DNS server is a critical target (5); attack is a reconnaissance scan (2); traffic is allowed to target (2); target is not patched (2)

9. Defensive Recommendations

Host based – The DNS server needs to be updated to the latest version of BIND – currently 8.2.2 patchlevel 5 (http://www.isc.org/products/BIND/bind8.html)
Network Based – Block the initiating source and monitor for further activity.

10. Sample Test Question

DNS queries use which protocol?

A) ICMP
B) UDP
C) TCP
D) Both UDP and TCP, depending on the size of the request

Correct Answer: D

Trace Two

```
[**] SCAN - Whisker Stealth Mode 8- Start Stop Web access attempt [**]
07/24-10:30:27.807498 10.0.0.154:1306 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:483  DF
*****PA* Seq: 0xF633E2BD   Ack: 0x1861A9   Win: 0x7D78

[**] SCAN - Whisker Stealth Mode 8- cfappman access attempt [**]
```

```
07/24-10:30:27.836459 10.0.0.154:1307 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:489  DF
*****PA* Seq: 0xF6846849   Ack: 0x1A55CF   Win: 0x7D78

[**] IIS-scripts-browse [**]
07/24-10:30:27.891920 10.0.0.154:1310 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:507  DF
*****PA* Seq: 0xF620DCC0   Ack: 0x1A5631   Win: 0x7D78

[**] IIS-carbo.dll [**]
07/24-10:30:28.315875 10.0.0.154:1329 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:620  DF
*****PA* Seq: 0xF605F6A3   Ack: 0x1A575B   Win: 0x7D78

[**] IDS219 - WEB-CGI-Perl access attempt [**]
07/24-10:30:28.405273 10.0.0.154:1335 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:656  DF
*****PA* Seq: 0xF6C49E6B   Ack: 0x1A57C5   Win: 0x7D78

[**] IDS219 - WEB-CGI-Perl access attempt [**]
07/24-10:30:28.416841 10.0.0.154:1336 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:662  DF
*****PA* Seq: 0xF6A2B03D   Ack: 0x1A57EB   Win: 0x7D78

[**] SCAN - Whisker Stealth Mode 8- Web Distribution access attempt
[**]
07/24-10:30:28.558115 10.0.0.154:1346 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:722  DF
*****PA* Seq: 0xF6063D12   Ack: 0x1A58B7   Win: 0x7D78

[**] SCAN - Whisker Stealth Mode 8- Handler CGI access attempt [**]
07/24-10:30:28.572520 10.0.0.154:1347 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:728  DF
*****PA* Seq: 0xF66D3976   Ack: 0x1A5865   Win: 0x7D78

[**] SCAN - Whisker Stealth Mode 8- wrap CGI access attempt [**]
07/24-10:30:28.584085 10.0.0.154:1348 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:734  DF
*****PA* Seq: 0xF63AC34A   Ack: 0x1A5889   Win: 0x7D78

[**] SCAN - Whisker Stealth Mode 8- Mall log order access attempt [**]
07/24-10:30:28.624496 10.0.0.154:1351 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:752  DF
*****PA* Seq: 0xF5F15C17   Ack: 0x1A5891   Win: 0x7D78

[**] SCAN - Whisker Stealth- Shopping cart access attempt [**]
07/24-10:30:28.653251 10.0.0.154:1353 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:764  DF
*****PA* Seq: 0xF5DDE2B0   Ack: 0x1A58A5   Win: 0x7D78

[**] SCAN - Whisker Stealth Mode 8- Order log access attempt [**]
07/24-10:30:28.684932 10.0.0.154:1355 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:776  DF
*****PA* Seq: 0xF5F3A2A5   Ack: 0x1A58CB   Win: 0x7D78

[**] SCAN - Whisker Stealth- BigConf access attempt [**]
07/24-10:30:28.702165 10.0.0.154:1356 -> 192.168.38.50:80
```

```
TCP TTL:63 TOS:0x0 ID:782  DF
*****PA* Seq: 0xF5F98B8C   Ack: 0x1A58E3   Win: 0x7D78

[**] WEB-CGI-rwwwshell CGI access attempt [**]
07/24-10:30:29.084640 10.0.0.154:1378 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:914  DF
*****PA* Seq: 0xF656279D   Ack: 0x1A5A66   Win: 0x7D78

[**] WEB-CGI-rwwwshell CGI access attempt [**]
07/24-10:30:29.099049 10.0.0.154:1379 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:920  DF
*****PA* Seq: 0xF6025126   Ack: 0x1A5A84   Win: 0x7D78
```

1. Source of the trace:

My Network

2. Detect

This detect was generated by Snort version 1.6 using the 06082k base rule set from http://www.snort.org.

3. Probability the source address was spoofed:

This is unlikely since the attack runs over TCP and requires a three-way handshake. The address may have been spoofed, but the attacker would not retrieve any information from the intended target.

4. Description of attack:

Snort detected a Whisker scan using “Windows \ delimiter” IDS evasion technique. This scan is designed to search through a web server for possible vulnerabilities while incorporating certain attributes that can cause an IDS to ignore the scan.

5. Attack Mechanism

Two of the biggest holes found in web servers are vulnerable CGI-BIN scripts and the msdacs.dll hole found in Windows IIS. Whisker is a scanner developed by Rain Forest Puppy (http://www.wiretrip.net) design to be the “next generation” of web server vulnerability scanners. In addition to scanning intelligently (i.e. not running CGI scans when no CGI-BIN directory exists), Whisker adds IDS evasion techniques to limit the amount of alarms set off by the scan. So instead of Whisker just using plain text HTTP GET requests, Whisker, among other things, will URL encoded all or part of the request to break up the literal plaintext string, such as /cgi-%62in/ph%66. It keeps the string-matching/packet-grep IDS systems from getting a positive id. For example, a Real Secure 3.1 sensor did not detect the above attack because of Whisker’s evasion techniques.

6. Correlations

Attacks targeting this type of target are a part of the SANS Top Ten vulnerabilities list (http://www.sans.org/topten.htm). Microsoft address the vulnerability in MDAC with their advisory 98-004 and 99-025. The mdac hole is also listed as CVE-1999-1011.

7. Evidence of Active Targeting

There is a good deal of evidence for active targeting here. The attacker not only knew which IP was a web server, but also knew it was running IIS. This can be seen by the exploits scanned for and the "Windows \ Delimiter" evasion technique used.

8. Severity

Severity = (Criticality + Lethality) - (System Countermeasures + Network Countermeasures)
(5 + 2) – (2 + 4) = 1

Server is a critical target (5); attack is a reconnaissance scan (2); traffic is allowed to target (2); target is patched (4)

9. Defensive Recommendations

Host based - None. The server has been properly patched for all known Windows NT and IIS exploits.
Network Based – Block the initiating source and monitor for further activity.

10. Sample Test Question

The following Snort alert is most likely targeting which operating system:

```
[**] IIS-scripts-browse [**]
07/24-10:30:27.891920 10.0.0.154:1310 -> 192.168.38.50:80
TCP TTL:63 TOS:0x0 ID:507  DF
*****PA* Seq: 0xF620DCC0   Ack: 0x1A5631   Win: 0x7D78
```

A) Linux
B) FreeBSD
C) Windows NT
D) Solaris

Correct answer: C

Trace Three

```
16:20:08.804322 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
```

```
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804391 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804459 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804528 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804598 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804666 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804735 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804804 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
16:20:08.804872 P steelrain.badguy.com > protect-65.goodguy.com: (frag
1109:9@65520)
                   4500 001d 0455 1ffe ff01 aa1c 0a00 0087
                   c0a8 2241 0800 0000 0000 0000 0000 0000
                   0000 0000 0000 0000 0000 0000 0000
```

1. Source of the trace:

My Network

2. Detect

This attack was detected by Snort version 1.6. The Snort “minfrag” preprocessor was set to 512 bytes. This trace is provided by tcpdump version 3.5.

3. Probability the source address was spoofed:

It is very likely the source address of the attack was spoofed. The attacker does not require any information back from a target, and therefore has need to use his legitimate IP address.

4. Description of attack:

This attack is a Denial-of-Service (DoS) that exploits a flaw in the Windows family of operating systems. The affected operating systems contain a flaw in the code that performs IP fragment reassembly. If a continuous stream of fragmented IP datagrams with a particular malformation were sent to an workstation, it would spend most or all of its CPU availability to process the fragments. The data rate needed to completely deny service varies depending on the machine and network conditions, but in most cases even relatively moderate rates would suffice. The exploit script "jolt2.c" (ftp://ftp.technotronic.com/denial/jolt2.c) has been circulated on the Internet and would cause these alerts.

5. Attack Mechanism

The vulnerability results because Windows 95, Windows 98, Windows NT 4.0 and Windows 2000 do not correctly perform IP fragment reassembly. If a stream of IP fragments containing a particular malformation were received, even at a relatively low rate, it could cause an affected machine to dedicate most or all of its CPU time to handling them.

6. Correlations

This attack was described in Microsoft advisory MS99-029. The exploit code was obtained from http://packetstorm.securify.com/0005-exploits/jolt2.c. This attack was originally published by the BindView's Razor team at http://razor.bindview.com/publish/advisories/adv_Jolt2.html.

7. Evidence of Active Targeting

There is evidence of active targeting here. The attacker aimed this exploit directly at a Windows NT domain controller, indicating the attacker knew what machine to target.

8. Severity

Severity = (Criticality + Lethality) - (System Countermeasures + Network Countermeasures)
(5 + 4) – (1 + 4) = 4

The domain controller is a critical target (5); attack is a highly effective DoS simply because it generates so many packets (4); traffic of this type is allowed onto the network (1); the system has been patched (4)

9. Defensive Recommendations

Host based - None. The server has been properly patched for all known Windows NT exploits.
Network Based – A firewall could be used to stop this attack, but filtering on a particular higher-level protocol might not be effective. The malformed fragments can arrive via any higher-level protocol. However, many networks filter for fragmented datagrams, and such a firewall would protect the machines behind it.

10. Sample Test Question

In the trace show above, what does the value "1109" represent?

A) The ICMP header size
B) The fragment offset
C) The fragment size
D) The fragment ID

Correct Answer: D

Trace Four

```
16:38:18.539599 P ip_hl < 5 (0)
                   4000 0028 b5b1 0000 4004 4000 8f9e 1f36
                   c0a8 2201 0000 0050 0000 0000 0000 0000
                   5000 2000 fe18 0000 0000 0000 0000
16:38:22.516507 P ip_hl < 5 (0)
                   4000 0028 b5b2 0000 4004 4000 8f9e 1f36
                   c0a8 2202 0000 0050 0000 0000 0000 0000
                   5000 2000 fe17 0000 0000 0000 0000
16:38:26.900242 P ip_hl < 5 (0)
                   4000 0028 b5b3 0000 4004 4000 8f9e 1f36
                   c0a8 2203 0000 0050 0000 0000 0000 0000
                   5000 2000 fe16 0000 0000 0000 0000
16:38:31.520997 P ip_hl < 5 (0)
                   4000 0028 b5b4 0000 4004 4000 8f9e 1f36
                   c0a8 2204 0000 0050 0000 0000 0000 0000
                   5000 2000 fe15 0000 0000 0000 0000
16:38:35.542084 P ip_hl < 5 (0)
                   4000 0028 b5b5 0000 4004 4000 8f9e 1f36
                   c0a8 2205 0000 0050 0000 0000 0000 0000
                   5000 2000 fe14 0000 0000 0000 0000
16:38:39.446439 P ip_hl < 5 (0)
                   4000 0028 b5b6 0000 4004 4000 8f9e 1f36
                   c0a8 2206 0000 0050 0000 0000 0000 0000
                   5000 2000 fe13 0000 0000 0000 0000
16:38:43.007138 P ip_hl < 5 (0)
                   4000 0028 b5b7 0000 4004 4000 8f9e 1f36
                   c0a8 2207 0000 0050 0000 0000 0000 0000
                   5000 2000 fe12 0000 0000 0000 0000
16:38:46.866601 P ip_hl < 5 (0)
                   4000 0028 b5b8 0000 4004 4000 8f9e 1f36
                   c0a8 2208 0000 0050 0000 0000 0000 0000
                   5000 2000 fe11 0000 0000 0000 0000
16:38:50.464319 P ip_hl < 5 (0)
                   4000 0028 b5b9 0000 4004 4000 8f9e 1f36
                   c0a8 2209 0000 0050 0000 0000 0000 0000
                   5000 2000 fe10 0000 0000 0000 0000
```

1. Source of the trace:

My Network

2. Detect

This detect was generated by tcpdump version 3.5 using a filter looking for packets with the second byte (byte 1, the 4 bit header length) set to any hex value other than 5.

3. Probability the source address was spoofed:

It is very likely the source address of the attack was spoofed. The attacker does not require any information back from a target, and therefore has need to use his legitimate IP address.

4. Description of attack:

TCPDump version 3.4a has a bug that could allow a malicious user to craft a packet and stop the TCPDump from running. By decoding the hex output from TCPDump (version 3.5), the destination addresses are a protected class C network. The fourth octet increments by one with each packet. In the trace above, the first target is 192.168.34.1 and the last address is 192.168.34.9. It appears as though an attacker is attempting to shut down devices in promiscuous mode, such as an IDS, on a class C.

The packet creation utility apsend (http://www.elxsi.de) can create these types of packets by using the switch "-tcpd".

5. Attack Mechanism

On receiving an IP packet of Ipv4 and a header length of zero, tcpdump enters an infinite loop within the procedure ip_print() from file print_ip.c This happens because the header length (ihl) equals '0' and tcpdump tries to print the packet. The result is tcpdump crashing. Certain other software that looks at packets using a device in promiscuous mode has also been found to be vulnerable to this attack. Specifically, Ethereal up through version 0.8.8 will crash when it receives such a packet.

6. Correlations

This attack was described in BugTraq Message-ID: 86010116595204.00853@dune.

7. Evidence of Active Targeting

There is no evidence of active targeting here. An attacker used a exploit aimed at an entire class C, not a specific machine, in the hopes of hitting a vulnerable box.

8. Severity

Severity = (Criticality + Lethality) - (System Countermeasures + Network Countermeasures)
(5 + 2) – (1 + 4) = 1

An IDS is a critical target (5); attack is a not likely to succeed (2); traffic of this type is allowed onto the network (1); no IDS on the network is running a vulnerable version of tcpdump (5)

9. Defensive Recommendations

Host based - None. There are no vulnerable system running.
Network Based – Block packets with an IP header length less than 20.

10. Sample Test Question

Which byte of the IP header contains the packet length?

A) Byte 0
B) Byte 1
C) Byte 9
D) Byte 13

Correct Answer: B

Trace Five

```
12:49:53.808975 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                  0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                  00fe ffff ffff ffff 0000 0000 0340 1848
                  0140 2cf7 ffbf f6b0 0840 0000 0100
12:49:55.819026 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                  0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                  00fe ffff ffff ffff 0000 0000 0340 1848
                  0140 2cf7 ffbf f6b0 0840 0000 0100
12:49:57.829091 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                  0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                  00fe ffff ffff ffff 0000 0000 0340 1848
                  0140 2cf7 ffbf f6b0 0840 0000 0100
12:49:59.839149 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                  0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                  00fe ffff ffff ffff 0000 0000 0340 1848
                  0140 2cf7 ffbf f6b0 0840 0000 0100
12:50:01.849222 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                  0001 0800 0604 0002 0060 8cc8 8cf7 0a00
```

```
                   00fe ffff ffff ffff 0000 0000 0340 1848
                   0140 2cf7 ffbf f6b0 0840 0000 0100
12:50:03.859277 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                   0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                   00fe ffff ffff ffff 0000 0000 0340 1848
                   0140 2cf7 ffbf f6b0 0840 0000 0100
12:50:05.869320 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                   0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                   00fe ffff ffff ffff 0000 0000 0340 1848
                   0140 2cf7 ffbf f6b0 0840 0000 0100
```

1. Source of the trace:

My Network

2. Detect

This detect was generated by tcpdump version 3.5 using a filter looking for the gateway address within a protected domain.

3. Probability the source address was spoofed:

In this network trace, the source IP address, gw.badguy.com is definitely spoofed. The attack works by spoofing a real address. However, the MAC address, 0:60:8c:c8:8c:f7, is not spoofed. This box is controlled by an attacker.

4. Description of attack:

This trace shows an ARP poisoning attack in progress. An attacker is broadcasting bogus ARP replies as though they were coming from the actual network gateway, except the attacker is using his own MAC address. This type of attack allows a malicious user to sniff packets on a switch segment other than their own.

This ARP poisoning utility is a port of the dsniff suite by Dug Song available from http://www.monkey.org/~dugsong/dsniff/. A detailed explanation of this attack is shown in Section Two.

5. Attack Mechanism

This attack relies on crafted ARP packets to intercept traffic from a target host or network intended for another host, usually a gateway. It does this by forging ARP replies. An attacker sends a storm of forged ARP packets indicating the attacker's system has the hardware address of the network gateway. Since these types of ARP packets are sent to broadcast, the attacker's system will be used by a target host as the network gateway. If this succeeds, the victim system will route its traffic to the attacker's system. Under normal circumstances, a host will send an ARP request to the broadcast address asking for the hardware address of the network gateway. Since the attacker is flooding the switch with forged ARP replies, the host receives the forged ARP reply before the actual network gateway has the opportunity to respond. Should the actual gateway reply with its hardware address, it will be ignored by the target host since it already has received a response to its ARP query.

6. Correlations

This ARP cache poisoning attack was described in the Vuln-Dev mailing list on Security Focus. The message ID is: Message-ID: <613309F30B6DD2118C020000F809376C018BC8E9@emss03m09.orl.lmco.com>.

7. Evidence of Active Targeting

Obviously, there has not only been active targeting here, but previous access. An attacker or malicious user has gained control of an workstation within the protected domain.

8. Severity

Severity = (Criticality + Lethality) - (System Countermeasures + Network Countermeasures)
(5 + 5) – (1 + 1) = 8

Target is the entire network (5); attack can sniff all traffic on a switch (5); traffic of this type (ARP) is not usually monitored on the internal network (1); a host has already been compromised (1)

9. Defensive Recommendations

Scan the protected network for sniffers using L0pht's AntiSniff to detect devices in promiscuous mode. Track the MAC address to its source and begin a Incident Response recovery.

10. Sample Test Question

Which of the following statements is TRUE about ARP packet?

A) ARP packets cannot pass routers
B) ARP packets are the primary packets found on the Internet
C) ARP packets cannot be forged
D) ARP packets are only used by Windows NT

Correct Answer: A

Section Two – Evaluate An Attack

A Review and Evaluation of "ARPREDIRECT"

1. Introduction

This section of the practical assignment evaluates an attack. The attack I chose to evaluate is the "arpredirect" utility found in Dug Song's dsniff suite (available at http:// naughty.monkey.org/~dugsong//dsniff/). This evaluation includes description of how the attack would actually work in a real-word environment. It also shows packet traces of the attack in action.

1.1 Background

Over the past several years, intruders have relied heavily on "sniffing" to gain unauthorized access to various network resources. By sniffing network traffic as it goes by on the wire, an intruder can read various types of data. This data can be as simple as e-mail messages or as potentially dangerous as usernames and passwords. In the past, the only way to defeat this type of attack was to use encryption. For those organizations that could not afford to use encryption, a modest amount of defense against sniffers has been the use of network switches.

An intelligent hub with basic routing capabilities is known as a *switch*.[1] Switches can read the destination address of packets, filter the packets, and forward them as appropriate. While a hub will transmit a packet to all devices on the hub, regardless of where the packet is actually destined, a switch will only broadcast data to the segment of the switch where the target is actually located. This ensures that data will only go where it is intended. By doing this, a switch ensures that a sniffer must be located on that particular destination segment to be effective. A sniffer could also be effective if it is physically placed on the spanning port on a switch, a specific port which sees all switch traffic. This situation, however, rarely happens due to malicious circumstances. While this does not completely erase the threat of a sniffer, it ensures that sniffing is too impractical to be used by anyone but insiders.

Recently, a new suite of utilities have been introduced that makes sniffing through switches possible, if not entirely practical. Written by Dug Song at CITI, the Center for Information Technology Integration, a research lab at the University of Michigan[2], the "dsniff" tool suite has introduced a method for circumventing normal switching protections by manipulating the Address Resolution Protocol (ARP). This technique has the possibility of eliminating any security benefit gained from using a switch instead of a hub. The result of this would mean an intruder could sniff a particular segment on a switch that would normally be unavailable.

[1] *Communications Systems and Networks.* Horak, Ray. M&T Books, Foster City, CA, 2000.

[2] http:// naughty.monkey.org/~dugsong//dsniff/

1.2 Overview

Although there is no way to completely ensure computer networks are safe from sniffer attacks, there are measures that can be taken which minimize the likelihood of a sniffer attack succeeding. This report starts with a review of how sniffer attacks through switches actually work. Once we understand how the attack works, we will review how we detect this type of attack. For detection to be effective, we must be able to detect the attack before it occurs as well as the attack as it is occurring. Finally, we will look at ways to prevent an attack from succeeding.

2. The Advantages Of Switched Networks

Switched networks have become popular, especially is small to medium size enterprises, for good reason. A switched network has several advantages over a broadcast network. The two main areas are performance and security.

2.1 Performance advantages

One of the primary advantages of using a switched network in lieu of a broadcast network is performance. A broadcast network, in short, is a simple collision domain where any packet bound for any destination on the network is actually sent to every destination on the network. A switched network on the other hand has the advantage of "learning" the network. By this, we mean that a switch will receive a packet and will only send it to the port where the actual destination resides. This lowers the amount of inconsequential traffic on the wire. In addition to this, a switch has memory buffers used to store native data packets, examine them for errors, then fragment them into smaller pieces. These fragments are then passed across a shared bus on the switch and directed to the appropriate output port. At the receiving port, the fragments gather in a memory buffer and are reformed into a reconstructed packet. Depending on the switch, the data fragment size may be as little as 28 bytes or as large as 4,096 bytes. The trade-off in size is recouped in performance – small fragments enable more users to share the bus at any given time, while larger fragments improve switching speed since the must analyze and act on fewer packet headers[3]. As with anything, there is an exception to this rule. If an ARP packet with the destination hardware address of ff:ff:ff:ff:ff:ff (broadcast) is sent out, this ARP packet will be sent to all segments of the switch.

2.2 Security Advantages

The primary security advantage to a switched network involves defeating a sniffer. Since the switch only sends traffic to its intended destination without passing information to other workstations on a network. Typically, in a switched environment, a compromised host with a sniffer installed would only see network traffic bound for that particular segment where the sniffer resides. The data on the other segments, by nature, will be masked from the compromised segment.

[3] *Communications Systems and Networks.* Horak, Ray. M&T Books, Foster City, CA, 2000.

3. Sniffing On A Switch

In order to sniff through a switch, a two-part attack must take place. In the first stage of sniffing, an attacker must convince a victim host to forward their traffic back to him. This phase is know as "ARP Poisoning". In the second phase of sniffing, the attacker must forward the victim's traffic back to its intended destination as though nothing unusual has happened. This phase is known as "IP Forwarding". We will begin by looking at the technical details of "ARP Poisoning".

3.1 ARP Poisoning

As we have already established, switching technology routes packets from one destination to another without passing them by any of the other stations on a network. This in turn reduces the risk of the packets being sniffed by an attacker. However, if an attacker uses the arpredirect utility included with Dug Song's "dsniff" tool suite, sniffing through switches becomes possible.

The actual mechanics of the attack are relatively simple. An attacker sends a storm of forged ARP packets indicating the attacker's system has the hardware address of the network gateway. Since these types of ARP packets are sent to broadcast, the attacker's system will be used by a target host as the network gateway as mentioned in the exception in section 2.1. If this succeeds, the victim system will route its traffic to the attacker's system. Under normal circumstances, a host will send an ARP request to the broadcast address asking for the hardware address of the network gateway. Since the attacker is flooding the switch with forged ARP replies, the host receives the forged ARP reply before the actual network gateway has the opportunity to respond. Should the actual gateway reply with its hardware address, it will be ignored by the target host since it already has received a response to its ARP query. A packet trace of this process is shown below:

1. Attacker is shown broadcasting forged ARP replies. The attacker's hardware address 0:60:8c:c8:8c:f7.

```
12:49:53.808975 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
              0001 0800 0604 0002 0060 8cc8 8cf7 0a00
              00fe ffff ffff ffff 0000 0000 0340 1848
              0140 2cf7 ffbf f6b0 0840 0000 0100
12:49:55.819026 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
              0001 0800 0604 0002 0060 8cc8 8cf7 0a00
              00fe ffff ffff ffff 0000 0000 0340 1848
              0140 2cf7 ffbf f6b0 0840 0000 0100
12:49:57.829091 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
              0001 0800 0604 0002 0060 8cc8 8cf7 0a00
              00fe ffff ffff ffff 0000 0000 0340 1848
              0140 2cf7 ffbf f6b0 0840 0000 0100
12:49:59.839149 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
              0001 0800 0604 0002 0060 8cc8 8cf7 0a00
              00fe ffff ffff ffff 0000 0000 0340 1848
              0140 2cf7 ffbf f6b0 0840 0000 0100
12:50:01.849222 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
              0001 0800 0604 0002 0060 8cc8 8cf7 0a00
              00fe ffff ffff ffff 0000 0000 0340 1848
```

```
                   0140 2cf7 ffbf f6b0 0840 0000 0100
      12:50:03.859277 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                   0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                   00fe ffff ffff ffff 0000 0000 0340 1848
                   0140 2cf7 ffbf f6b0 0840 0000 0100
      12:50:05.869320 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                   0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                   00fe ffff ffff ffff 0000 0000 0340 1848
                   0140 2cf7 ffbf f6b0 0840 0000 0100
```

2. A victim host sends an ARP request asking for the hardware address of the network gateway. The target host's hardware address is 0:60:8c:c8:f7.

```
      12:50:06.179416 arp who-has gw.badguy.com tell target.badguy.com
      (0:50:f:e:e1:f1)
                   0001 0800 0604 0001 0060 8cc8 8cf7 0a00
                   0028 0000 0000 0000 0a00 00fe
```

3. The forged ARP reply is received by the victim host before the true gateway can respond. The victim host caches the hardware address and forwards its traffic to this address.

```
      12:50:06.869320 B arp reply gw.badguy.com is-at 0:60:8c:c8:8c:f7
                   0001 0800 0604 0002 0060 8cc8 8cf7 0a00
                   00fe ffff ffff ffff 0000 0000 0340 1848
                   0140 2cf7 ffbf f6b0 0840 0000 0100
```

3.2 IP Forwarding

Now that the attacker's system has the victim traffic, he must get the traffic to its intended destination as though nothing has happened[4]. Should the attacker choose not to forward the traffic to its intended destination, an attackers can effectively perform a denial-of-service attack on the targets. The IP forwarding is done so that the end user still gets the information requested and notices nothing out of the ordinary. There are two methods of accomplishing this task. The first method involves using kernel-level IP forwarding on a Linux system. In truth, any operation system that can perform IP forwarding is acceptable, but Linux is the easiest to configure for this purpose. This option is set in Linux by /proc/sys/net/ipv4/ip _forward[5] and allows a host to act as a gateway.

The other method available for performing IP forwarding is through the use of a fragrouter on a system to perform the IP forwarding. Using this method, the fragrouter program itself forwards the sniffed traffic back to its intended destination[6]. While most intrusion detection systems will notice fragmented packets created by fragrouter, when run in "Normal IP Forwarding" mode, there is no discernable difference in the traffic.

Once the traffic is received by an attacker, any information desired can be gained. As we have already stated, this can include information from telnet, FTP, POP (post office

[4] "Switched Networks Lose Their Security Advantage Due to Packet Capturing Tool". Stuart McClure and Joel Scambray, *InfoWorld Magazine*, IDG Publications, May 26, 2000.

[5] *Running Linux, 3rd Edition.* Matt Welsh, Matthias Kalle Dalheimer & Lar Kaufman, O'Reilly, August, 1999.

[6] http://www.anzen.com/research/nidsbench/

protocol), HTTP, IRC (Internet Relay Chat), and many others session. Information collected from these sessions could contain passwords, user names, e-mail messages, web request information, etc.

3.3 Worst Case Scenario

Extensive laboratory test has shown one crucial fact – the ARP packets needed to execute this attack will not pass through a router. For example, switches on the other sides of a router cannot be sniffed without first compromising additional workstations. This limits the amount of potential damage that can be caused from this form of sniffing. This attack would only be useful after an attacker has already gained access to a host on a protected site. Once illicit access has been achieved, an attacker would then be able to sniff other segments of the protected domain using poisoned ARP packets. Also, an insider could use this attack to sniff other sections of the network normally protected by switches. An attacker would not be able to use this attack to directly sniff a protected network from the internet without compromising a protected host. A network compromised in this manner will also suffer some performance issues. This is because there is now twice as much traffic on the network: the traffic going to the attacker's forwarding device, and that traffic going to its intended destination.

4. Detection And Countermeasures

4.1 Detecting an Active Sniffer

Using network based intrusion detection system (IDS) to detect this attack would be impractical. This is because the traffic associated with this attack should never pass an IDS sensor on the network perimeter, since the ARP packets do not cross routers. Sensors cannot detect what it cannot see. The network-based IDS could only detect the attack if the sensor resided on the switch segment being attacked. The key to detection is monitoring ARP packets. The attack succeeds by broadcasting an enormous amount of ARP packets across the network. Therefore, monitoring the ARP traffic on a network for such ARP broadcast storms is the simplest way to detect this particular attack. One tool to monitor ARP traffic is arpwatch by Craig Leres[7]. This tool would allow network administrators and information assurance personnel to monitor their internal networks for unusually large amounts of ARP traffic on the wire. Another solution for detection would be to run multiple sensors at various locations (i.e. behind every route point on the network) on the protected network. This increases the chance that the ARP traffic would be seen by an IDS sensor. This too could prove unpractical due to the amount of sensors needed to fully cover a typical switched network.

There is also a product that could be used for detecting a sniffer from LØpht Heavy Industries called AntiSniff[8]. This product works by running a number of non-intrusive tests which can determine whether or not a system is listening to all network communications. This makes is possible to remotely detect the passive act of

[7] ftp://ftp.ee.lbl.gov/arpwatch.tar.Z

[8] http://www.l0pht.com/antisniff/overview.html

eavesdropping on network communications. AntiSniff will even detect packet sniffers installed by a rogue insider who may have legitimate administrative access to a machine, but still should not be monitoring all network traffic. While this product will not help defeat a sniffer as one is installed, if run regularly, it will detect an active sniffer on any given network.

4.2 Defeating an Active Sniffer

Unfortunately, the only true way to defeat this form of sniffing is by using point-to-point encryption. This is the only countermeasure that will ensure traffic is only able to be read by its intended recipient. While this method will not actually defeat the sniffer, it will ensure that the data collected by a sniffer is unreadable. At this time, encryption of this sort is not required by the most organizations. As encryption technologies become more practical and attacks become more sophisticated, requiring point-to-point encryption beyond a base perimeter should become mandatory. The other obvious defeating tactic is to ensure that workstations are regularly updated with appropriate security patches. Hardening workstations in this manner will help ensure they are not compromised in the first place. Again, this attack will not succeed without some type of prior access.

5. Conclusions

While this attack certainly removes the security advantages from a switched network, it will only be effective from within the protected network. Switches will still maintain their advantage over hubs due their performance advantages. Since this is an insider or after access attack, little needs to be done to strengthen network perimeter defenses. However, this attack once again points out the need for dedicated, point-to-point encryption. It also points out the need for administrators to keep up with patches that will harden the workstations on the network. By doing this, it will reduce the likelihood that a workstation will be compromised. In turn, this reduces the likelihood a sniffer can be installed. Defensive measures and constant vigilance are the only way to ensure traffic on a protected network is never sniffed.

Section Three – Analyze This

John Dough
Bogus Corporation
123 Financial District

Otherplace, CA, 90210

Your Ref	Your Letter Dated	Our Ref	Date
IDS Bid	July 15, 2000	IDS Bid	7/31/00

Attn: John Dough

Mr Dough:

Re: IDS Bid

Our consulting service, Eager Beaver Consulting, has been asked to provide a bid to your corporation for security services for your location. To this end, we have received and reviewed 50 files produced by your Snort intrusion detection system (IDS). You will find the analysis of these files in the paragraphs below. Unfortunately, I do not know anything about the network being monitored. Therefore, several assumptions about the network must be made. For example, I am assuming that certain IPs are critical targets (i.e. E-mail or DNS servers). I am also assuming no one from the monitored network should be performing any scanning or malicious activities. Since the IP addresses have been sanitized with MY.NET, it will be difficult to tell which alerts are "wrong numbers". For example, Keesler Air Force Base has a domain of 158.157. The Silesian Technical University of Poland has a domain of 157.158. These domains have often been mis-typed, causing many false-positive alerts.

The table below is a summary of alerts for your location for the timeframe of 5/16/00 through 6/23/00. The total number of alerts during this timeframe was 86,971. However, the actual number of alerts should be much higher. The data received by Eager Beaver Consulting was fragmented at best. Many days of alerts were not available. Also, many days had identical files, and thus identical alerts. This does not compromise the integrity of the data provided, but it also does not give a thorough look at the network security posture of your location.

Alert Name	*Number of Alerts*	*Number of Sources*	*Number of Destinations*
WinGate 8080 Attempt	40375	351	20412
SYN-FIN scan!	18182	15	16057
Watchlist 000220 IL-ISDNNET-990517	8544	30	25
Watchlist 000222 NET-NCFC	6843	32	18
WinGate 1080 Attempt	3563	330	1012
Attempted Sun RPC high port access	2914	11	10
NMAP TCP ping!	2778	7	755

SUNRPC highport access!	1948	12	729
SNMP public access	804	22	1
SMB Name Wildcard	321	16	8
Null scan!	212	92	77
GIAC 000218 VA-CIRT port 34555	158	31	14
Tiny Fragments - Possible Hostile Activity	157	3	3
GIAC 000218 VA-CIRT port 35555	107	29	15
Probable NMAP fingerprint attempt	29	6	17
External RPC call	17	3	3
GIAC 08-feb-2000	12	1	1
Happy 99 Virus	2	2	2
Queso fingerprint	1	1	1

Problem Area One

The first thing discovered were a lot of alerts for the following rules:

Watchlist 000220 IL-ISDNNET-990517

Watchlist 000222 NET-NCFC

This is a problem, because I cannot find a rule that would cause this alert. I have reviewed the 06082k and 07122k rules from http://www.snort.org and can find no reference to any rule that may have cause these alerts. I have also reviewed the rules from Max Vision (http://www.whitehats.com) and can find no reference to either rule. Here is a summary of what I can ascertain from these alerts:

Alert Name	*Number of Alerts*	*Number of Sources*	*Number of Destinations*
Watchlist 000220 IL-ISDNNET-990517	8544	30	25
Watchlist 000222 NET-NCFC	6843	32	18

For the Watchlist 000220 IL-ISDNNET-990517 rules, all alerts come from IPs within the 212.179.xxx.xxx range. This range belongs to Arel-Net out of Israel. I can only assume previous activity has been seen from this source and a rule has been written for monitoring further activity. Most of the activity has source and destination ports above

1024. This activity occurs during the entire monitoring timeframe, with the earliest alert at 00:00:04.418704 on 05/16 and the last alert at 04:47:40.464245 on 06/23.

For the Watchlist 000222 NET-NCFC rules, all alerts come from IPs with the 159.226.xxx.xxx range. This range belongs to the Computer Network Center Chinese Academy of Sciences out of Beijing, China. Again, I can only assume previous activity has been seen from this source and a rule has been written for monitoring further activity. Most of the connections are to destination port 25 (smtp) on MY.NET.6.47. This appears to be the e-mail server, as there are many connections to this port from various sources. For the most part, none of the connections appear malicious beyond their source address. This activity also occurs during the entire monitoring timeframe, with the earliest such alert at 00:56:56.473342 on 05/16, and the last such alert at 10:20:08.833794 on 06/23.

Problem Area Two - Reconnaissance Scanning

The Snort logs show a great deal of reconnaissance scanning being done against the protected network. The scans could indicate precursors to future attacks. A breakout of these scans is shown in the table below:

Alert Name	*Number of Alerts*	*Number of Sources*	*Number of Destinations*
WinGate 8080 Attempt	40375	351	20412
SYN-FIN scan!	18182	11	15933
NMAP TCP ping!	3183	7	958
Null scan!	138	66	53
Probable NMAP fingerprint attempt	35	5	16
Queso fingerprint	1	1	1

WinGate 8080 Attempt

As seen in the chart above, this alert accounts for roughly 45% of the total number of alerts for the reviewed timeframe. Of these 40,375 alerts, 22,144 of them can be attributed to a single IP: 202.38.128.188. This IP performed a sequential can hitting every IP from MY.NET.1.0 through MY.NET.254.255. This IP is registered to the Institute of High Energy Physics located in Beijing, China. This is a part of the Chinese Academy of Sciences - also seen in the "Watchlist 000222 NET-NCFC" alerts. It appears the monitored network is a favorite target of this institution. Unless there is legitimate activity expected between the monitored network and China, I would recommend search the Asia Pacific Network Information Centre (APNIC http://www.apnic.net) for all IP ranges associated with the China Academy of Sciences and block them at the network perimeter.

The other alerts, however, appear to be users from an outside network using monitored proxy servers on MY.NET for illegitimate purposes. There are a total of six machines that appear to be allowing anonymous proxy from outside networks through. These machines are:

MY.NET.253.105 (The big winner with 13,819 alerts)
MY.NET.99.85
MY.NET.97.11
MY.NET.97.203
MY.NET.97.108
MY.NET.97.69

Of these IPs, MY.NET.253.105 has the most traffic flowing through it. This IP has most likely been posted to an Internet site that lists anonymous proxy servers. These machines need to be locked down. They may have been set up as anonymous proxy servers without the knowledge of system administrators. Obviously, this is a wide-spread problem that needs serious attention.

SYN-FIN Scanning

The biggest SYN-FIN reconnaissance offender is the IP 204.60.176.2. This IP belongs to Southern New England Telephone out of Meriden, CT and is responsible for 13,562 of the SYN-FIN scan alerts. This IP scanned - sequentially - every IP from 1 through 255 starting with the subnet MY.NET.1.xxx while incrementing - sequentially - the third octet from 1 through 255. This activity occurred on 6/13 and lasted about 20 minutes.

Another 4,594 alerts on SYN-FIN scanning come from 142.250.225.137. This IP is registered to the University of Toronto, Canada. This scan was somewhat more selective, which could indicate some previous targeting activity. This scan occurred on 5/22 and again lasted about 20 minutes.

The other SYN-FIN scans were much more selective, with each source only sending seven packets or less to specific port on specific targets. The table below shows a quick summary of these scans.

Source IP	***Number of Alerts***	***Number of Targets***	***Destination Ports***
210.222.31.100 (Korea Network Information Center)	7	2	1524, 2222
210.118.8.50 (Standard Network System Inc, Seoul, Korea)	6	3	109
155.230.152.165 (Kyungpook National	6	2	53

University, Taegu, Korea)			

It is interesting that all of these are from Korea. This could possibly be an indication of distributed scanning, but there is not enough evidence to confirm this.

False-Positives

Because of some of the limiting factors of the alert files (i.e. no TCP flags shown), very few alerts can be confirmed as false-positives. The other alerts for SYN-FIN do not show any pattern or coherence, so I will think of them as false-positives. However, one IP from Demon Internet in the UK, 194.217.242.41 can be termed as a false-positive with some degree of assurance based on previous activity seen from this service provider.

There are several other alerts listed as part of reconnaissance scans, but none were significant enough warrant further investigation at this time. Should Eager Beaver win this contract, every scan will be dealt with appropriately.

Problem Area Three - Possible Compromised Hosts

During the course of reviewing these alerts, one box stood out for activities outbound from the monitored network. This machine was MY.NET.253.12. It caused 8,426 alerts going to other IPs within the monitored domain. A summary of these alerts is shown in the table below:

Alert Name	***Number of Alerts***	***Number of Destinations***
Null scan!	22	6
Probable NMAP fingerprint attempt	21	13
WinGate 1080 Attempt	1856	926
WinGate 8080 Attempt	1858	925
SUNRPC highport access!	1908	953
NMAP TCP ping!	2761	952

As you can see, this IP was scanning for proxy servers and running SunRPC programs. Since there are a large number of NMAP TCP ping alerts associated with this activity, we can assume the user is running Nmap (http://www.insecure.org) to accomplish this. There are two probable reasons for this activity: the legitimate user for this IP is performing malicious activities, or the workstation has been compromised and is now being used as a jumping off point for further penetration. Since the only alerts with MY.NET.253.12 as a destination came as a part of a scan to the entire domain, we can safely say the machine was not compromised during the monitoring timeframe. However, the machine could have been compromised previous to monitoring. It would be best to begin a forensics analysis of this machine to determine if it has been compromised or if a legitimate user has been abusing the network.

There were also two alerts for the "Happy 99" virus. This worm will open a window entitled "Happy New Year 1999 !!" and shows a firework display to disguise its installation. This worm sends itself to other users when the infected computer is online. These alerts were targeted towards MY.NET.253.51 and MY.NET.253.52. These machines have possibly been compromised by this virus. A complete recovery of these machines needs to occur before any information on the workstations can be considered trustworthy.

A large number of machines on your network appear to be infected with a derivative of the Back Orifice virus. A total of 265 alerts were seen on ports 34555 and 35555. As described by a VA-CIRT member (http://www.sans.org/y2k/021800.htm), these ports have been used by derivatives of Back Orifice to communicate with other systems. A total of fifteen workstations were seen with this type of activity. Unfortunately, due to the lack of correlation in the port scan files, we at Eager Beaver cannot be certain of the activity on these machines. You should treat the workstations as infected and begin recovery procedures. These workstations need to be sanitized and recovered before they are placed back on the network.

Problem Area Four - System Misconfigurations

The primary concern for configuration problems deals with IP MY.NET.101.192. There are a total of 804 SNMP public access alert all headed to this machine. The source for all of these alerts appears to be MY.NET.97.xxx. Even if this is allowed, expected traffic, it is extremely insecure. It also concerns Eager Beaver that this traffic is being seen by you IDS sensor. If your sensor is one the perimeter of your network, that means this traffic is also going out to the perimeter of your network. Of course, since we have no topology map or description of your network, until you bring Eager Beaver in to do a thorough analysis, we will not know for sure.

Several system also appear to allow NetBIOS services. Specifically MY.NET.100.130 is repeatedly targeted by NetBIOS Name Service (port 137) connections. If this is a domain controller, your Windows networks have allowed a huge amount of information gathering to occur. All NetBIOS ports should be prevented from crossing a network perimeter.

In a similar vein, a few machines appear to be running SunRPC services. Specifically, MY.NET.6.15 showed many attempts to port 111 (sunrpc). These services, if not properly patched, are the quickest way to having root access on that box.

Recommendations

As you can see from a sample of our analysis above, our complete analysis services can be very detailed and thorough. We will ensure you data is protected by the best scripts in the known universe. By hiring our consulting company, the following things will be provided in order to ensure network security at your location:

1. Define a Security Policy

Our consulting corporation will provide your location with a clear, concise, and well defined security policy. Since you have sent none to us, we must assume that you do not have one. A good security policy will enable you to design and implement a good security program that addresses the entire security infrastructure, build user awareness, plan for technical solutions to security problem, and set solid boundaries for user activities.

2. Reliable Sensors

The biggest problem with the data provided is its inconsistency. The first action to resolve this is to purchase machines capable of processing the data. These machines need to be on uninterruptible power supplies (UPS) and physically secured in a locked room. This will help ensure the data on the sensors does not become compromised. It will also help ensure there are no gaps in the recording of data.

3. Regular Review of Data

As you can see from our sample analysis above, six weeks worth of data can be immense on a network of your size, even if less than half of the data is available. Eager Beaver will provide you with a daily analysis of all alerts produced by the Snort IDS. This will also make it much easier to respond to an incident should a compromise occur.

4. Adaptive Firewalls

Based on the amount of alerts and scans received, your location does not have an effective firewall in place. Eager Beaver will provide firewall administration that includes taking real-time data from the Snort sensor to update firewall rules on the fly. This will be accomplished through a free utility known as Guardian (http://www.chaotic.org/~astevens/Guardian/). This will help deal with the tremendous amount of port scans and reconnaissance that is targeted towards your network.

5. Vulnerability Scans

Eager Beaver will perform a network vulnerability assessment on your network to help lock down the holes that currently exist. This will ensure that YOU know what all is going on inside your network. It will also give you a look at how an intruder would see you network. This effort would be done without the knowledge of system administrators in order to gauge their skill at responding to an active attack.

6. Training

Eager Beaver would provide you with both general user training and specific system administrator training. This will greatly increase the level of knowledge and security awareness at your location. Always remember, a secure network is a team effort.

The total cost for all of these services, plus travel and expenses, comes to $3,250,417.34 per year for the next four years. Please contact us at your earliest convenience to arrange a payment play.

Sincerely,

Dustin C. Childs
Eager Beaver Consulting