Global Information Assurance Certification Paper

GIAC Intrusion Analyst Certification Practical Submittal

SANS 2001 New Orleans
Roderick Campbell

Submitted April 4, 2001

Assignment 1 – 5 Detects With Analysis

Detect 1:

This trace was taken from my home network.

```
[**] MISC Large ICMP Packet [**]
02/28-17:15:05.457879 attacker.net.20 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:5185 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO

[**] MISC Large ICMP Packet [**]
02/28-17:20:51.930597 attacker.net.20 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:52497 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO

[**] MISC Large ICMP Packet [**]
02/28-17:28:00.495439 attacker.net.21 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:37121 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO

[some deleted for brevity]

[**] MISC Large ICMP Packet [**]
03/01-12:59:58.852529 attacker.net.20 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:33345 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO

[**] MISC Large ICMP Packet [**]
03/01-13:04:59.574213 attacker.net.21 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:12865 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO

[**] MISC Large ICMP Packet [**]
03/01-13:10:08.611759 attacker.net.20 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:51985 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO
```

Detect Generation:

This detect was generated by Snort v1.7 (www.snort.org).

Description of Attack:

This attack uses large ICMP Echo Request packets (a.k.a. ping) for possibly one of two reasons. It is either an attempt to ping flood the target host, thereby decreasing it's available bandwidth (and on slow Internet connections, such as dialup, can render the target unreachable) or it is an attempt to discover the MTU of the target host in an information reconnaissance effort.

Attack Mechanism:

Each of the packets received in this attack was 1500 bytes long, which is the maximum frame length (maximum transmission unit, or MTU) for ethernet. These packets were crafted in order for the attacking host to try and discover the MTU of the network the target host resides on.

[sample packet]

```
[**] MISC Large ICMP Packet [**]
02/28-17:15:05.457879 attacker.net.20 -> my.net.210
ICMP TTL:246 TOS:0x0 ID:5185 IpLen:20 DgmLen:1500 DF
Type:8  Code:0  ID:39612   Seq:57072  ECHO
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

[rest of packet deleted for brevity]

As you can see from the above packet, the DgmLen is 1500 bytes; the byte of the packet payload are stuffed with 00 (null). This is an obvious indication of packet crafting.

Another (and probably the strongest) reason why this looks like an MTU discovery attack, is that the DF (Don't Fragment) bit is set. By setting this bit in the IP header and using large packets, the attacker expects to receive ICMP messages from my gateway router when fragmentation had to occur but couldn't due to the DF bit set. By starting with a large datagram length, and incrementally decreasing this value, the attacker hopes to discover the target's MTU in a reconnaissance effort that might later be used in a later attack. (To date, there has not been another attack detected from this host.)

I first mentioned that because of the large packet size, this could possibly have been a ping flood attack. However, look at the following summary of the detects:

```
02/28-17:15:05.457879 attacker.net.20 -> my.net.210
02/28-17:20:51.930597 attacker.net.20 -> my.net.210
02/28-17:28:00.495439 attacker.net.21 -> my.net.210
03/01-10:10:03.428732 attacker.net.20 -> my.net.210
03/01-10:17:41.281981 attacker.net.21 -> my.net.210
03/01-10:23:56.158280 attacker.net.20 -> my.net.210
03/01-10:33:02.175043 attacker.net.20 -> my.net.210
03/01-10:38:47.880678 attacker.net.20 -> my.net.210
03/01-10:44:33.526701 attacker.net.21 -> my.net.210
03/01-11:07:36.219193 attacker.net.21 -> my.net.210
03/01-11:19:07.649781 attacker.net.20 -> my.net.210
03/01-11:26:22.621532 attacker.net.21 -> my.net.210
03/01-11:41:24.720941 attacker.net.20 -> my.net.210
03/01-11:49:24.628131 attacker.net.21 -> my.net.210
03/01-12:04:43.433171 attacker.net.20 -> my.net.210
03/01-12:19:45.444026 attacker.net.21 -> my.net.210
03/01-12:24:46.159101 attacker.net.21 -> my.net.210
03/01-12:29:47.129597 attacker.net.20 -> my.net.210
03/01-12:39:48.816926 attacker.net.21 -> my.net.210
03/01-12:44:49.552899 attacker.net.20 -> my.net.210
03/01-12:59:58.852529 attacker.net.20 -> my.net.210
03/01-13:04:59.574213 attacker.net.21 -> my.net.210
03/01-13:10:08.611759 attacker.net.20 -> my.net.210
```

By looking at the timestamps, you can see that the detects are not coming in rapid bursts, which disproves the theory that this could be a ping flood. You should notice, however, that they do come in fairly regular intervals. The detects seem to come from the two separate alternating hosts in 10 minute intervals. (The intervals are not perfect, and this is probably due to Snort dropping packets.)

For all these reasons and more, it is my opinion that this is an MTU discovery attack using large datagrams with the DF bit set.

Probability of Spoofed Source Address:

This source address is probably not spoofed. Because this attack is appears to be a reconnaissance effort at MTU discovery, the attacker will want to receive ICMP replies and thus not spoof his address.

Correlation:

Traces from tcpdump confirmed this traffic (and also the packet loss experienced with snort), but did not provide any other insight into the attack.

The firewall log (ipchains log on Linux) did not provide any correlation of this traffic. This alarmed me at first, however after checking the filter rules on the firewall, I realized that these type of ICMP packets (echo requests) are not logged (denied, but not logged) for the purpose of controlling logfile size (echo requests are all too common).

Evidence of Active Targeting:

This attack was actively targeted at the host. Two consecutive IP addresses on the same subnet (leading me to assume one attacker with two computers) soliciting echo replies with large datagrams at regular intervals over a 20 hour timeframe leads me to believe this host was actively targeted.

Severity:

(Criticality + Lethality) – (System Countermeasures + Net Countermeasures) = Severity

Criticality	5	This host is the firewall between my internal network and the Internet.
Lethality	1	The lethality of this attack is low, as it was simply a reconnaissance attempt. However, follow-up attacks from this could be more lethal. However, no further attacks were detected.
System Countermeasures	5	Current operating system, with kernel and services operating at current patch levels. Minimal services accessible.
Net Countermeasures	3	A somewhat permissive firewall (NOTE: the target under attack here is the firewall), however the key is that the firewall does not respond to the attack.
Severity	**-2**	

Defensive Recommendation:

No immediate defensive recommendations. However, since this was a reconnaissance effort, this host should be monitored for further reconnaissance attempts and also further attacks resulting from the reconnaissance efforts.

Multiple Choice Question:

Question: What about the above detect strongly suggests this attack is an MTU discovery?

A) ICMP ID field is unchanging

B) Null bytes in the packet body
C) DF bit set
D) Unique IP ID numbers

Answer: C

Detect 2:

This detect was taken from my home network. NOTE: Only snippets of the traces are show here, as all of the alerts generated by this attack exceed 5MB. (Timestamps may be slightly out of order, as these were grouped by message type, however in the alert file, they were all interspersed.)

```
03/17-19:36:07.957006 attacker.net.55 -> my.net.210
[**] spp_portscan: PORTSCAN DETECTED from attacker.net.55
(THRESHOLD 4 connections exceeded in 1 seconds) [**]
[**] spp_portscan: portscan status from attacker.net.55: 17
connections across 1 hosts: TCP(17), UDP(0) [**]
[**] spp_portscan: portscan status from attacker.net.55: 18
connections across 1 hosts: TCP(18), UDP(0) [**]
[**] spp_portscan: portscan status from attacker.net.55: 20
connections across 1 hosts: TCP(20), UDP(0) [**]
[**] spp_portscan: portscan status from attacker.net.55: 25
connections across 1 hosts: TCP(25), UDP(0) [**]
[**] spp_portscan: portscan status from attacker.net.55: 22
connections across 1 hosts: TCP(22), UDP(0) [**]
[**] spp_portscan: portscan status from attacker.net.55: 24
connections across 1 hosts: TCP(24), UDP(0) [**]

[snip]

[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-20:10:55.023262 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:23902 IpLen:20 DgmLen:68
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:27444
UDP TTL:116 TOS:0x0 ID:12316 IpLen:20 DgmLen:40
Len: 20
** END OF DUMP

[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-20:10:55.026142 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:23903 IpLen:20 DgmLen:68
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:27444
UDP TTL:116 TOS:0x0 ID:12313 IpLen:20 DgmLen:40
Len: 20
** END OF DUMP
```

```
[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-20:10:55.028640 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:23904 IpLen:20 DgmLen:68
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:27444
UDP TTL:116 TOS:0x0 ID:12310 IpLen:20 DgmLen:40
Len: 20
** END OF DUMP
```

[snip]

```
[**] ICMP Echo Reply [**]
03/17-20:10:55.020945 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:12318 IpLen:20 DgmLen:28
Type:0  Code:0  ID:3  Seq:1280  ECHO REPLY

[**] ICMP Echo Reply [**]
03/17-20:10:55.022430 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:12317 IpLen:20 DgmLen:33
Type:0  Code:0  ID:567  Seq:1024  ECHO REPLY

[**] ICMP Echo Reply [**]
03/17-20:10:55.023883 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:12315 IpLen:20 DgmLen:28
Type:0  Code:0  ID:3  Seq:768  ECHO REPLY
```

[snip]

```
[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-20:45:06.658945 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:27885 IpLen:20 DgmLen:70
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:2049 -> my.net.210:34555
UDP TTL:116 TOS:0x0 ID:20853 IpLen:20 DgmLen:42
Len: 22
** END OF DUMP

[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-20:45:06.662494 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:27887 IpLen:20 DgmLen:70
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:2049 -> my.net.210:34555
UDP TTL:116 TOS:0x0 ID:20849 IpLen:20 DgmLen:42
Len: 22
** END OF DUMP
```

```
[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-20:45:06.666310 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:27889 IpLen:20 DgmLen:70
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:2049 -> my.net.210:34555
UDP TTL:116 TOS:0x0 ID:20845 IpLen:20 DgmLen:42
Len: 22
** END OF DUMP
```

[snip]

```
[**] ICMP Echo Reply [**]
03/17-20:45:06.659579 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:20852 IpLen:20 DgmLen:28
Type:0  Code:0  ID:3  Seq:1280  ECHO REPLY

[**] ICMP Echo Reply [**]
03/17-20:45:06.660331 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:20851 IpLen:20 DgmLen:33
Type:0  Code:0  ID:567  Seq:1024  ECHO REPLY

[**] ICMP Echo Reply [**]
03/17-20:45:06.662852 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:20848 IpLen:20 DgmLen:28
Type:0  Code:0  ID:3  Seq:768  ECHO REPLY
```

[snip]

```
[**] DDOS shaft handler to agent [**]
03/17-21:05:10.365652 attacker.net.55:53 ->
my.net.210:18753
UDP TTL:116 TOS:0x0 ID:43619 IpLen:20 DgmLen:49
Len: 29

[**] DDOS shaft handler to agent [**]
03/17-21:05:10.366892 attacker.net.55:53 ->
my.net.210:18753
UDP TTL:116 TOS:0x0 ID:43620 IpLen:20 DgmLen:49
Len: 29

[**] DDOS shaft handler to agent [**]
03/17-21:05:10.592117 attacker.net.55:53 ->
my.net.210:18753
UDP TTL:116 TOS:0x0 ID:43621 IpLen:20 DgmLen:49
Len: 29

[**] DDOS shaft handler to agent [**]
03/17-21:05:10.592857 attacker.net.55:53 ->
my.net.210:18753
UDP TTL:116 TOS:0x0 ID:43622 IpLen:20 DgmLen:49
Len: 29
```

[snip]

```
[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-21:11:16.649869 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:32482 IpLen:20 DgmLen:68
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:27444
UDP TTL:116 TOS:0x0 ID:63010 IpLen:20 DgmLen:40
Len: 20
** END OF DUMP

[snip]

[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-21:11:14.451885 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:32477 IpLen:20 DgmLen:70
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:34555
UDP TTL:116 TOS:0x0 ID:62872 IpLen:20 DgmLen:42
Len: 22
** END OF DUMP

[snip]

[**] DDOS shaft handler to agent [**]
03/17-21:11:40.964265 attacker.net.55:53 ->
my.net.210:18753
UDP TTL:116 TOS:0x0 ID:64464 IpLen:20 DgmLen:49
Len: 29

[snip]

[**] ICMP Echo Reply [**]
03/17-21:11:40.957376 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:64458 IpLen:20 DgmLen:28
Type:0  Code:0  ID:3  Seq:51979  ECHO REPLY
```

Detect Generation:

This detect was generated by Snort v1.7.

INTERESTING NOTE: The alerts that piqued my attention to this attack were not the actual UDP packets themselves, but rather the ICMP Port Unreachable messages sent back by my firewall; an "inverse detect" if you will.

Description of Attack:

This is a trace of a combination attack (and not a very well researched one, I might add). It starts off with a simple TCP portscan (with several more TCP and UDP portscans following throughout the duration of the attack, not shown). Two different DDoS attacks were used throughout the duration of the detect (Trinoo, Shaft).

Attack Mechanism:

The attack starts off with an initial TCP port scan of the target host. This is presumably an initial reconnaissance effort for the attacks to follow later, however it probably would have been better for the attacker to start off with a UDP portscan, as all of the DDoS attacks he used utilize UDP ports.

The first grouping of ICMP messages are being sent back to the attacking host, informing it that UDP port 27444 is not available. UDP/27444 is a well known port for the DDoS tool Trinoo.

The next grouping shows several ICMP Echo Replies. What interested me in these, is that they were unsolicited replies. My first instinct was that these were being used as a decoy; to hide the Trinoo traffic by filling the sensor logs with echo replies (more on this later). These echo replies were interspersed throughout the entire duration of the attack.

The next grouping shows more ICMP Port Unreachable messages, this time being sent back to the attacker because of a UDP attempt to 34555. UDP/34555 is a well known port for the Windows version of Trinoo.

Again, more unsolicited echo replies.

Next we see that Snort DID detect the signature for Shaft (another DDoS tool). This was the only attack that Snort detected directly. (The others were caught indirectly, by seeing all of the Port Unreachable messages headed towards the attacking host.)

The rest are merely more snippets from the very end of the attack. As you can see, this was a very determined (or perhaps just bored) attacker, to have initiated the attack at 19:36, and carried it through until 21:11.

More on those unsolicited echo replies… Initially, I thought they were merely decoys, to try and hide his other traffic in the sensor logs. However, upon further examination, I found this:

```
[**] ICMP Echo Reply [**]
03/17-20:10:55.024628 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:12314 IpLen:20 DgmLen:33
Type:0  Code:0  ID:567  Seq:512  ECHO REPLY
31 32 33 34 35                                    12345


[**] ICMP Echo Reply [**]
03/17-21:11:40.952203 attacker.net.55 -> my.net.210
ICMP TTL:116 TOS:0x0 ID:64453 IpLen:20 DgmLen:33
Type:0  Code:0  ID:567  Seq:50699  ECHO REPLY
31 32 33 34 35                                    12345
```

This led me to believe that a TFN probe/attack was also underway. However, upon looking at the signatures from the Snort rules database (www.snort.org), this doesn't seem to match any of the TFN rules. I tried matching it up to other ICMP rules in the snort database, but still could not match this to a signature. It has me a little baffled. Perhaps it was just a decoy after all, or a strange ping flood.

What is significant about this attack, is not the tools used by the attacker (Trinoo and Shaft), but rather the sheer scale on which this attack was launched, and the length of time that it lasted. Especially considering that it was unsuccessful. The Port Unreachable messages shown above for attempts to ports UDP/34555 and UDP/27444 show that this Trinoo "master-to-daemon" attempt was unsuccessful. Likewise:

```
[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-21:11:37.626436 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:32503 IpLen:20 DgmLen:82
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:18753
UDP TTL:116 TOS:0x0 ID:64267 IpLen:20 DgmLen:54
Len: 34
** END OF DUMP

[**] ICMP Destination Unreachable (Port Unreachable) [**]
03/17-21:11:40.499860 my.net.210 -> attacker.net.55
ICMP TTL:255 TOS:0xC0 ID:32506 IpLen:20 DgmLen:77
Type:3  Code:3  DESTINATION UNREACHABLE: PORT UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
attacker.net.55:53 -> my.net.210:18753
UDP TTL:116 TOS:0x0 ID:64426 IpLen:20 DgmLen:49
Len: 29
** END OF DUMP
```

These traces show that the Shaft "handler-to-agent" attempts were also unsuccessful.

Another interesting point to be learned from this attack: ICMP can be used for inversely detecting attacks. In this case, the original Trinoo traffic was never detected by Snort, however the ICMP Port Unreachable messages drew my attention to the traffic, which uncovered a truly massive attack.

Probability of Spoofed Source Address:

It is well known that most DDoS attackers spoof their IP addresses. However, in this case, I am inclined to believe that the source address is real. Mainly because the source IP for all these detects is the same source IP from the initial portscan, and all the followup portscans detected throughout the duration of the attack. Portscans are most often reconnaissance attempts, which is what I believe they are here.

Correlation:

Data collected by tcpdump confirmed these detects, however did not provide any additional information about the echo replies.

"Defenses Against Distributed Denial of Service Attacks"
Gary C. Kessler
November 29, 2000
http://www.sans.org/infosecFAQ/threats/DDoS.htm

Evidence of Active Targeting:

Yes, this host was actively targeted. I believe that the time duration of the attack would indicate that this host was actively targeted.

Severity:

(Criticality + Lethality) – (System Countermeasures + Net Countermeasures) = Severity

Criticality	5	The targeted host is my firewall.
Lethality	1	This attack was very unlikely to succeed, as the daemons the attacker was attempting to contact did not exist on the target host.
System Countermeasures	5	Modern operating system, kernel and software at current patch levels.
Net Countermeasures	2	Fairly permissive firewall, but it was the firewall that was under attack. However it still was not susceptible to the attacks.
Severity	**-1**	

Defensive Recommendation:

This was an attack actively targeted at the host, where the attacker was expecting to find listening daemons for the Trinoo and Shaft DDoS attacks. The source address is not believed to be spoofed, so continue to monitor attacking host for further attempts, as well as perform periodic audits on the target system to make sure that the tools for these DDoS attacks (and others) are not present.

Multiple Choice Question:

Question: What is possibly being demonstrated in the first ICMP detects above?

A) Nothing, just normal ICMP traffic.
B) An inverse detect of malicious traffic.
C) The firewall is blocking outbound access to port 53 on remote hosts.
D) Host attacker.net.55 does not have a DNS server present.

Answer: B

Detect 3:

This detect was taken from my home network.

```
[**] DNS named iquery attempt [**]
03/18-05:46:20.478647 ATTACKER.NET.231:3844 -> MY.NET.210:53
UDP TTL:54 TOS:0x0 ID:17442 IpLen:20 DgmLen:55
Len: 35
B0 F2 09 80 00 00 00 01 00 00 00 00 00 00 01 00  ................
01 00 00 7A 69 00 04 04 03 02 01                 ...zi......
```

Detect Generation:

This detect was generated by Snort v1.7.

Description of Attack:

This is a probe, to discover if the target host is vulnerable to the inverse query exploit found in versions 4.9 and 8 of BIND, first published April 08, 1998.

Attack Mechanism:

The actual attack works by exploiting a boundary condition error found in the vulnerable versions of BIND on specific platforms. When processing an inverse query, BIND fails to do proper

bounds checking. When performing a memory copy, portions of the named exec can be overwritten and commands run on the host.

It is the binary contents of the flags in the DNS query packet which triggers this alert. (`09 80 00 00 00 01 00 00 00 00`)

In this case, no responses, either DNS or ICMP were sent back to the attacker. The target was not running BIND, and it drops all ingress port 53 packets in the packet filter

Probability of Spoofed Source Address:

This trace is that of a pre-attack probe, therefore the attacker needs to receive a response from the target host. Thus, it is unlikely that this address was spoofed.

Correlation:

CVE-1999-0009
ArachNIDS 277
BugTraq 134r

Evidence of Active Targeting:

There is no evidence of active targeting in this trace. Only one probe was detected, and there was no followup attack. Also, the target host was not running BIND, or any name service for that matter. There was no previous reconnaissance from this IP address.

Severity:

(Criticality + Lethality) – (System Countermeasures + Net Countermeasures) = Severity

Criticality	5	The target host is my firewall.
Lethality	0	The probe was not researched before hand, nor did it have any chance of returning a positive response.
System Countermeasures	5	Modern operating system, kernel and software at current patch levels.
Net Countermeasures	5	The packet filter on the firewall drops all ingress packets on port 53.
Severity	**-5**	

Defensive Recommendation:

No defensive actions are needed.

Multiple Choice Question:

Question: The above trace is an example of what?

A) A buffer overflow exploit.
B) A specific pre-attack probe.
C) A malformed UDP header.
D) A UDP scan for DNS servers.

Answer: B

Detect 4:

This detect was taken from my home network.

```
[**] ICMP PING NMAP [**]
03/04-17:22:33.291511 ATTACKER.NET.46 -> MY.NET.210
ICMP TTL:29 TOS:0x0 ID:29782 IpLen:20 DgmLen:28
Type:8  Code:0  ID:5223   Seq:0  ECHO

[**] SCAN nmap TCP [**]
03/04-17:22:34.138181 ATTACKER.NET.46:62069 ->
MY.NET.210:720
TCP TTL:26 TOS:0x0 ID:36978 IpLen:20 DgmLen:40
***A**** Seq: 0x5BE776FE  Ack: 0x0  Win: 0x800  TcpLen: 20

[**] SCAN nmap TCP [**]
03/04-17:22:34.140649 ATTACKER.NET.46:62069 ->
MY.NET.210:567
TCP TTL:26 TOS:0x0 ID:15475 IpLen:20 DgmLen:40
***A**** Seq: 0x5BE776FE  Ack: 0x0  Win: 0x800  TcpLen: 20

[**] SCAN nmap TCP [**]
03/04-17:22:34.146059 ATTACKER.NET.46:62069 ->
MY.NET.210:857
TCP TTL:26 TOS:0x0 ID:60929 IpLen:20 DgmLen:40
***A**** Seq: 0x5BE776FE  Ack: 0x0  Win: 0x800  TcpLen: 20

[snip]

[**] SCAN nmap TCP [**]
03/04-17:23:55.041909 ATTACKER.NET.46:62077 ->
MY.NET.210:21
TCP TTL:26 TOS:0x0 ID:32559 IpLen:20 DgmLen:40
***A**** Seq: 0x2F7C75B4  Ack: 0x0  Win: 0x800  TcpLen: 20

[**] SCAN nmap TCP [**]
03/04-17:23:55.042405 ATTACKER.NET.46:62077 ->
MY.NET.210:540
TCP TTL:26 TOS:0x0 ID:42040 IpLen:20 DgmLen:40
***A**** Seq: 0x2F7C75B4  Ack: 0x0  Win: 0x800  TcpLen: 20

[**] SCAN nmap TCP [**]
03/04-17:23:55.043658 ATTACKER.NET.46:62077 ->
MY.NET.210:123
TCP TTL:26 TOS:0x0 ID:25430 IpLen:20 DgmLen:40
***A**** Seq: 0x2F7C75B4  Ack: 0x0  Win: 0x800  TcpLen: 20
```

Detect Generation:

This detect was generated by Snort v1.7.

Description of Attack:

Snort generated an alert to this attack as a TCP scan from the tool Nmap. However, it was unlike any Nmap scan that I had previously seen. Instead of using a SYN-FIN scan, a half-open SYN

scan, or a vanilla port scan, here only ACK packets are sent to the target host. Also notice the random sequence in which destination ports are probed, and the semi-static sequence numbers.

Attack Mechanism:

Because Snort alerted this traffic as being generated by Nmap, I consulted the Nmap website (www.insecure.org/namp). According to the website, an ACK scan can be used to discover a firewall's ruleset, and whether or not it is stateful or merely filters incoming SYN connections.

> *This scan type sends an ACK packet (with random looking acknowledgement/sequence numbers) to the ports specified. If a RST comes back, the ports is classified as "unfiltered". If nothing comes back (or if an ICMP unreachable is returned), the port is classified as "filtered". Note that nmap usu- ally doesn't print "unfiltered" ports, so getting **no** ports shown in the output is usually a sign that all the probes got through (and returned RSTs).*

Note that the signature of this attack will supposedly only detect older versions of Nmap which set the ACK flag to 0. However, I was able to recreate this trace using Nmp-2.53.

Probability of Spoofed Source Address:

This is a reconnaissance attack. The attacker is expecting to receive information back as a result of this scan. Thus, this source address is most likely not spoofed.

Correlation:

ArachNIDS 28
www.insecure.org/nmap

Evidence of Active Targeting:

While host scans generally do not always signify active targeting, this situation is slightly different. Because ACK scanning can be used to map out a firewall's ruleset, and the target host in this trace is a firewall, this was a trace that was actively targeted.

Severity:

(Criticality + Lethality) – (System Countermeasures + Net Countermeasures) = Severity

Criticality	5	The target system is my firewall.
Lethality	3	While this was a reconnaissance attack, the information returned by it could be very valuable in a future attack. Especially since I was able to reproduce this trace using Nmap, and map out the ruleset on my firewall.
System Countermeasures	5	Current operating system running at current patch levels.
Net Countermeasures	2	A fairly permissive firewall, that did return information about itself.
Severity	**1**	

Defensive Recommendation:

This was a reconnaissance effort. The attacking system should be monitored for a followup attack that could result from this pre-attack probe.

Multiple Choice Question:

Question: What about the above trace differentiates it from other portscans?

A) The ACK field is always set to 0x0.
B) The destination ports are probed in random order
C) The sequence numbers are not unique for each connection.
D) It sends ACK packets to the target, instead of SYN or SYN-FIN.

Answer: D

Detect 5:

This detect was taken from my home network. It is another good example of an "inverse detect" by logging ICMP messages.

```
[**] ICMP Destination Unreachable (Undefined Code!) [**]
04/01-00:19:13.034436 REMOTE.NET.96 -> MY.NET.210
ICMP TTL:113 TOS:0x0 ID:10729 IpLen:20 DgmLen:56
Type:3  Code:13  DESTINATION UNREACHABLE: PACKET FILTERED
** ORIGINAL DATAGRAM DUMP:
MY.NET.210:6666 -> 0.0.0.0:1024
TCP TTL:10 TOS:0x0 ID:40216 IpLen:20 DgmLen:69
1*UAP*S* Seq: 0xCD070000  Ack: 0xDD4D899E  Win: 0xA2DF
TcpLen: 16  UrgPtr: 0x1583
** END OF DUMP

[**] ICMP Destination Unreachable (Undefined Code!) [**]
04/01-00:19:13.036900 REMOTE.NET.96 -> MY.NET.210
ICMP TTL:113 TOS:0x0 ID:10731 IpLen:20 DgmLen:56
Type:3  Code:13  DESTINATION UNREACHABLE: PACKET FILTERED
** ORIGINAL DATAGRAM DUMP:
MY.NET.210:6667 -> 0.0.0.0:1024
TCP TTL:10 TOS:0x0 ID:40216 IpLen:20 DgmLen:69
1*UAP*S* Seq: 0xCD070000  Ack: 0xDD4D899E  Win: 0xA2DF
TcpLen: 16  UrgPtr: 0x1583
** END OF DUMP

[**] ICMP Destination Unreachable (Undefined Code!) [**]
04/01-00:19:13.039843 REMOTE.NET.96 -> MY.NET.210
ICMP TTL:113 TOS:0x0 ID:10733 IpLen:20 DgmLen:56
Type:3  Code:13  DESTINATION UNREACHABLE: PACKET FILTERED
** ORIGINAL DATAGRAM DUMP:
MY.NET.210:6668 -> 0.0.0.0:1024
TCP TTL:10 TOS:0x0 ID:40216 IpLen:20 DgmLen:69
1*UAP*S* Seq: 0xCD070000  Ack: 0xDD4D899E  Win: 0xA2DF
TcpLen: 16  UrgPtr: 0x1583
** END OF DUMP
```

[snip]

```
[**] ICMP Destination Unreachable (Undefined Code!) [**]
04/01-00:19:34.140090 REMOTE.NET.96 -> MY.NET.210
ICMP TTL:113 TOS:0x0 ID:12885 IpLen:20 DgmLen:56
Type:3  Code:13  DESTINATION UNREACHABLE: PACKET FILTERED
** ORIGINAL DATAGRAM DUMP:
MY.NET.210:6666 -> 0.0.0.0:1292
TCP TTL:10 TOS:0x0 ID:40216 IpLen:20 DgmLen:69
1**APRSF Seq: 0xCD070000  Ack: 0xBAB7B6FC  Win: 0x85E
TcpLen: 16
** END OF DUMP

[**] ICMP Destination Unreachable (Undefined Code!) [**]
04/01-00:19:34.144026 REMOTE.NET.96 -> MY.NET.210
ICMP TTL:113 TOS:0x0 ID:12886 IpLen:20 DgmLen:56
Type:3  Code:13  DESTINATION UNREACHABLE: PACKET FILTERED
** ORIGINAL DATAGRAM DUMP:
MY.NET.210:6667 -> 0.0.0.0:1292
TCP TTL:10 TOS:0x0 ID:40216 IpLen:20 DgmLen:69
1**APRSF Seq: 0xCD070000  Ack: 0xBAB7B6FC  Win: 0x85E
TcpLen: 16
** END OF DUMP

[**] ICMP Destination Unreachable (Undefined Code!) [**]
04/01-00:19:34.146740 REMOTE.NET.96 -> MY.NET.210
ICMP TTL:113 TOS:0x0 ID:12887 IpLen:20 DgmLen:56
Type:3  Code:13  DESTINATION UNREACHABLE: PACKET FILTERED
** ORIGINAL DATAGRAM DUMP:
MY.NET.210:6668 -> 0.0.0.0:1292
TCP TTL:10 TOS:0x0 ID:40216 IpLen:20 DgmLen:69
1**APRSF Seq: 0xCD070000  Ack: 0xBAB7B6FC  Win: 0x85E
TcpLen: 16
** END OF DUMP
```

Detect Generation:

This detect was generated by Snort v1.7.

Description of Attack:

Again, this is a good example of an "inverse detect" from the ICMP messages. The original TCP datagrams (which are shown in each ICMP message) were not logged as being anomalous.

What exactly this attack is, I'm not entirely sure.

Attack Mechanism:

It appears that the traffic originates from my firewall, with a destination of 0.0.0.0; various destination TCP ports are used, and the source ports seem to stay within the 6666 – 6669 range. The traffic headed for 0.0.0.0 is received by REMOTE.NET.96, and an ICMP Destination Unreachable message is sent back to my firewall.

Checking the traffic captured by tcpdump during this time frame shows no initial egress traffic from my firewall to either 0.0.0.0 or REMOTE.NET.96. Is the remote host then completely fabricating these ICMP messages as well as the original datagram dumps within them? What would be the purpose or effectiveness of a technique like this?

Probability of Spoofed Source Address:

I cannot say for sure whether or not the source IP address was spoofed in this case, as I do not know what is being demonstrated in this trace. There is a high probability that the source address is spoofed, as there are obvious signs of packet craft. The original destination IP address is 0.0.0.0, and there are several different illegal TCP flag combinations shown in the original datagrams.

Correlation:

Without knowing for sure what this trace is trying to demonstrate, it is difficult to find appropriate correlating references. However, CVE-1999-0214 has a description of being able to cause a denial of service with forged ICMP unreachable packets.

Evidence of Active Targeting:

If this trace is a DoS as described by CVE-1999-0214, then that would be evidence of active targeting. However, there is no other evidence of active targeting in this case.

Severity:

(Criticality + Lethality) – (System Countermeasures + Net Countermeasures) = Severity

Criticality	5	This is my firewall that is being attacked.
Lethality	5	Because it is unknown what this trace is really demonstrating, I assign it a lethality of 5, just to be cautious.
System Countermeasures	5	Current operating system at current patch levels.
Net Countermeasures	2	A fairly permissive firewall ruleset.
Severity	**3**	

Defensive Recommendation:

The host REMOTE.NET.96 needs to be monitored closely for any other suspicious activity that might lend insight to the trace shown above. Corroboration from other security administrators would be helpful if they have seen the same types of traffic patterns.

Multiple Choice Question:

Question: Which of the following is anomalous with the traffic show above?

A) The destination host 0.0.0.0 in the orginal TCP datagram dumps.
B) The various TCP flag combinations shown in the original datagram dumps.
C) The fact that REMOTE.NET.96 answered this packet when it was sent to 0.0.0.0.
D) All of the above.

Answer: D

Assignment 2 – Research Essay

Event Correlation

A Limitation of Current Network Intrusion Detection Systems

Introduction

In today's modern business world, more and more information is being transmitted electronically over the corporate network, and an increasing amount of business is being conducted via the Internet. Protecting the information systems that conduct these transaction and disseminate information has never been more important that it is today. However, aside from protecting the individual hosts that reside on the network, monitoring the traffic produced and received by these hosts plays an invaluable role in network and information security. A network intrusion detection system (NIDS) is a vital component of a complete security policy.

Network intrusion detection systems work by analyzing the traffic travelling across the network, in either near real time or post-processing batches, looking for data or statistical patterns to indicate that a host is under attack, or that a protected enclave is being penetrated. Most NIDS use a method of detection called signature analysis. Signature analysis works by matching known vulnerability or penetration data patterns to the packet headers or payloads captured off the network. However, there are inherent problems with signature-based analysis: New exploits may not be matched against current known signatures, and existing exploits can be modified such that their signatures are different and thus evade detection. For these reasons, multiple approaches to network intrusion detection must be taken.

Nowadays, many sites incorporate multiple network intrusion sensors into their security policy, allowing for the strengths of some IDS to counterbalance the weaknesses of others. However, one of the major limitations in using multiple network intrusion sensors, is correlating the data to arrive at a complete and accurate conclusion and reduce false alarms.

Signature Based Analysis

The most common method of intrusion detection is signature analysis. Many commercial and freeware solutions are available that use attack signatures, or well known data patterns, for analysis. One such freeware tool is Snort (www.snort.org). An example of a signature for Snort is as follows:

```
alert tcp $EXTERNAL_NET 27374 -> $HOME_NET any (msg: "BACKDOOR SIG -
SubSseven 22"; flags: A+; content: "|0d0a5b52504c5d3030320d0a|";
reference:arachnids,485;)
```

This signature will cause Snort to generate an alert whenever a TCP packet from source port 27374 arrives bound for any of the local machines, with any destination port, has the ACK flag set, and the string `"|0d0a5b52504c5d3030320d0a|"` is found in the packet payload.

However, what happens if the attack tool is modified slightly so as to have a different signature? Chances are, it will evade detection by your NIDS.

How Criminals Defeat a NIDS

The most common method of intrusion detection is signature analysis, and so it logically follows that the most common way computer criminals evade detection is by leaving different signatures. It is easy enough to modify an attack tool such that the signature varies slightly; this slight variation has a good chance of evading detection by the NIDS. However, a modified signature will only remain undetected for so long, so computer criminals employ other techniques to avoid detection:

- Overwhelming an IDS
- Creating excessive false positives
- Using Unicode protocols to disguise signatures
- Compromising the IDS itself

NIDS are most effectively used at chokepoints in the network. Chokepoints are where an intrusion sensor will have the opportunity to examine the most traffic. The excessive amount of traffic arriving at a chokepoint, coupled with the speed at which it arrives places a terrible strain on the resources of the NIDS. By sending valid traffic, an attacker can increase the workload of a sensor in the hopes that it will begin dropping packets. Some of the packets that the NIDS drops could very well be the packets that you, as a security administrator, are most interested in and that the attacker does not want you to see.

Creating excessive false positives is a method similar to overwhelming the IDS, but it also can have a nasty side effect for the (semi) inept security administrator. An excessive amount of false positives could potentially cause the security administrator to disable that particular rule generating the false positives. If this happens, then a legitimate attack will sneak through undetected. Or even for the non-inept security administrator, excessive false positives can overwhelm the IDS by filling alert storage capabilities, so that alerts are no longer able to be logged.

Attackers can also use Unicode to disguise attack signatures. Like ASCII, Unicode maps characters to binary representation. Unlike ASCII, however, Unicode attempts to map every character of every language to a binary representation. To do this requires more

than the 256 characters available in the ASCII 8-bit map. Unicode uses a 16-bit mapping, and there are rules to extend even that.

Compromising the IDS itself is another tactic computer criminals use to avoid detection.

Why Multiple Sensors/Methods are Needed

Computer criminals have entirely too many tools at their disposal to avoid detection, especially by a signature-based NIDS. For this reason, multiple sensors which implement multiple methods of analysis should be used as part of a comprehensive security policy.

An example of multiple sensors might be to match Snort's signature matching capabilities with tcpdump's ability to capture every raw packet (without decoding and analysis) with relatively low loss. Not only is tcpdump likely to catch what Snort misses (tcpdump can also be used for signature based analysis after the fact), traffic captured by tcpdump can later be used to reconstruct an entire attack. Even if Snort missed the attack based on it's signature, Snort ICMP alerts may provide more information than first realized. (If you aren't logging ICMP alerts in Snort, you should be.) A failed attack will often result in the target host sending back an appropriate ICMP message to the attacker. These ICMP alerts can tip off the astute security administrator to go back and look at the traffic captured by tcpdump, and possibly even recreate the entire event! So what your signature-based IDS missed, tcpdump has captured. Multiple sensors have succeeded where a single sensor would have failed.

Stateful intrusion detection systems extend the capabilities of signature based analysis, because they are able to "remember" past packets. NIDS that use stateful protocol analysis are more adept at detecting slow port scans, and covert channel communications, whereas signature based analysis might not alert on such anomalous traffic. (In the case of a covert channel, stateful protocol analysis can alert when a threshold of unsolicited echo replies are received, such as in the case of Loki; and the stateful NIDS would know if echo replies were solicited or not.) This type of capability in a NIDS takes much of the work off the security administrator to pursue and analyze other attacks.

What Is Data/Event Correlation?

You say "tomayto" I say tomahto". Introducing multiple NIDS, possibly from multiple vendors, is a necessary component of any effective computer security policy. The only problem, though, is how to relate a detect from one sensor, to a detect made by another sensor. Logging formats and names for attacks vary from vendor to vendor; no one IDS speaks the same language. This lack of consistency makes the task of relating one event to another quite difficult.

Data collected by any IDS is multi-dimensional; a NIDS will generally log all the fields in the IP, TCP, UDP, and ICMP headers as well as the timestamp, and generate an alert if

an attack pattern is found. However, what the NIDS calls this attack may be completely different than what your other NIDS use to refer to the same attack. For example, one NIDS might call an out-of-band NetBIOS datagram a "WinNuke" attack, while yet others might call it a "NetBIOS OOB" attack; either way, they are referring to the same attack. It is left to the security administrator to examine all dimensions of the data collected from multiple sensors and conclude that this was the same attack. This type of manual event correlation is often an inefficient use of the security administrator's resources.

NIDS based event correlation is the core of an efficient and effective multiple network intrusion sensor deployment. It reduces the strain on the security administrator, and provides a more complete picture of an event, versus independent analysis of events gathered from multiple sensors.

The Common Vulnerability and Exposure Project

The Common Vulnerabilities and Exposures (CVE) is a dictionary (not a database) that provides standardized names for known information security vulnerabilities. The CVE dictionary maps attack patterns to a common name, so that alerts generated by disparate sensors use the same name to refer to an attack. This takes the strain of event correlation off the security administrator, provides better coverage across the network, easier interoperability between multiple vendors, and enhanced security.

The CVE, while a great starting point, is not a complete solution for event correlation. The CVE provides standardized names for known attack signatures. Thus, manual event correlation must still be performed by security administrators as new anomalous traffic patterns are discovered in the network, to which your multiple sensors refer differently.

Conclusion

Event correlation is a crucial aspect to any NIDS deployment, however it is currently one of the largest limitations in intrusion detection systems. A single intrusion sensor/analysis method is an ineffective way to monitor network traffic, as it is all too easy for attackers to evade an intrusion detection system; thus multiple sensors, which employ multiple analysis methods are needed. However, without event correlation, multiple sensors have not greatly increased the effectiveness of the security policy. The CVE attempts to remedy some of the limitations of event correlation in current intrusion detection solutions, however the CVE is still subject to the same limitations of signature based analysis. A common language is needed to refer to anomalous traffic that does not match current attack signatures, in order to extend event correlation beyond it's current limitations.

References

"Intrusion Detection, Take Two"; Network Computing; November 15, 1999
www.networkcomputing.com/1023/1023f1side2.html

"What difficulties are associated on matching events with attacks. Why is event/data correlation important?"; Guy Bruneau, GCIA; SANS Institute Intrusion Detection FAQ
www.sans.org/newlook/resources/IDFAQ/matching.htm

"CVE, The Key to Information Sharing"; The Common Vulnerabilities and Exposures Project
cve.mitre.org/about/introduction.html

"How Computer Criminals Defeat Intrusion Detection Systems"; Carolyn Meinel, eBiz
www.messageq.com/security/meinel_3.html

"What Are the Significant Gaps and Promising Future Directions?"; State of the Practice of Intrusion Detection Technologies; Carnegie Mellon Software Engineering Institute
www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028chap03.html

Assignment 3 – "Analyze This"

GIAC Enterprises:

Thank you for the opportunity to assess your current network security vulnerabilities. What follows is a summary analysis, based on the data provided. It should be noted, however, that the data set is not complete for the time-span given; disk failure, storage limitations, and/or power outages prevented data collection throughout the entire time period.

Introduction

The data provided for this analysis was from alert logs generated by the Snort intrusion detection system. A total of 194,039 alerts were reported, excluding portscans logged by the preprocessor. The preprocessor logged a total of 30,599 portscans.

Statistical analysis was used on these alerts to identify the most commonly targeted internal hosts, the most common attackers, the internal scanning hosts, the most common alerts generated, and the attacking hosts that have been watchlisted. Below you will find the results of these analyses, including references and recommended actions where applicable.

Internal Portscanning Hosts

Excessive portscans by internal hosts may be an indication of a compromised host that an attacker is using to launch portscans to other hosts on your network, or other hosts on the internet. Presented below are the five most signicant hosts conducting portscans on the network:

Host	Portscans	% of Portscans
MY.NET.217.150	6290	20.6%
MY.NET.217.158	4935	16.1%
MY.NET.100.230	3009	9.8%
MY.NET.219.126	2203	7.2%
MY.NET.253.24	2002	6.5%

As the figures show, these hosts homed on the private enclave are generating a higher than tolerable amount of portscans, thereby suggesting they have most likely been compromised. An audit of these systems should be performed immediately to determine if they have in fact been compromised. (If these systems have not been compromised, the security administrator should explain acceptable use to the users of these systems.)

In the likely event that these systems have been compromised, they should be removed from the network immediately, until the pertinent data can be backed up, and the operating system and softwares can be reinstalled and brought up to current patch levels.

In addition, the security administrator should explain acceptable use to the users of these systems.

Heavily Targeted Hosts

Heavily targeted hosts are those that have received a significantly higher percentage of the attacks than the other hosts on the network. Why these hosts are immensely popular with attackers should be investigated. (However, they are probably some of the common servers for the enclave, such as web, mail, and DNS, which are common targets of attacks.) If the heavily targeted hosts are client machines, the security administrator needs to immediately investigate why they are so poplular. Presented below are the five most heavily targeted hosts:

Host	Alerts	% of Alerts
MY.NET.201.222	37609	19.4%
MY.NET.220.126	25183	13.0%
MY.NET.225.234	9314	4.8%
MY.NET.1.3	5452	2.8%
MY.NET.202.94	5253	2.7%

Heavily Attacking Hosts

Heavily attacking hosts are those foreign hosts which generated a significantly higher percentage of alerts/attacks, exlcuding watchlist alerts, directed to hosts homed in the private enclave. Further monitoring of these hosts is required, and if malicious activity continues, they should be reported to the GIAC as possible watchlist candidates. Presented below are the five most active targeting hosts:

Host	Alerts	% of Alerts
212.179.79.2	48786	25.1%
212.179.27.111	39015	20.1%
211.34.40.1	17604	8.8%
209.67.50.203	16132	8.3%
195.56.182.206	9878	5.1%

212.179.79.2 **huc.ac.il**

inetnum: 212.179.79.0 - 212.179.79.63
netname: CREOSCITEX
descr: CREOSCITEX-SIFRA
country: IL
admin-c: ZV140-RIPE
tech-c: NP469-RIPE
status: ASSIGNED PA
notify: hostmaster@isdn.net.il
changed: hostmaster@isdn.net.il 20001109
source: RIPE

person: Zehavit Vigder
address: bezeq-international
address: 40 hashacham
address: petach tikva 49170 Israel
phone: +972 52 770145
fax-no: +972 9 8940763
e-mail: hostmaster@bezeqint.net
nic-hdl: ZV140-RIPE
changed: zehavitv@bezeqint.net 20000528
source: RIPE

person: Nati Pinko
address: Bezeq International
address: 40 Hashacham St.
address: Petach Tikvah Israel
phone: +972 3 9257761
e-mail: hostmaster@isdn.net.il
nic-hdl: NP469-RIPE
changed: registrar@ns.il 19990902
source: RIPE

Reference: www.ripe.net

212.179.27.111 **clnt-27111.bezeqint.net**

inetnum: 212.179.27.96 - 212.179.27.127
netname: INOBIZ
descr: INOBIZ-LAN
country: IL
admin-c: NP469-RIPE
tech-c: NP469-RIPE
status: ASSIGNED PA
notify: hostmaster@isdn.net.il
changed: hostmaster@isdn.net.il 20000106
source: RIPE

person: Nati Pinko
address: Bezeq International
address: 40 Hashacham St.
address: Petach Tikvah Israel
phone: +972 3 9257761
e-mail: hostmaster@isdn.net.il
nic-hdl: NP469-RIPE
changed: registrar@ns.il 19990902
source: RIPE

Reference: www.ripe.net

211.34.40.1 **(unresolved)**

```
inetnum 211.32.0.0 - 211.39.255.255
netname KRNIC-KR-22
descr KRNIC
descr Korea Network Information Center
country KR
admin-c WK1-AP, inverse
tech-c SL119-AP, inverse
remarks KRNIC Allocation Block remarks Authoritative Information
regarding assignments and
remarks allocations made from within this block can also be
remarks queried at whois.nic.or.kr
mnt-by APNIC-HM, inverse
mnt-lower MNT-KRNIC-AP, inverse
changed hostmaster@apnic.net 19990827
source APNIC

person Weon Kim, inverse
address Korea Network Information Center (KRNIC)
address ********** Important Notice **********
address KRNIC is the National Internet Registry.
address If you want to find detail assignment information
address about above IP address, please use http://whois.nic.or.kr
address ************************************************
address Narajongkeum B/D 14F, 1328-3, Seocho-dong, Seocho-Ku
address Seoul, 137-070, Republic of Korea
phone +82-2-2186-4500
fax-no +82-2-2186-4496
country KR
e-mail hostmaster@nic.or.kr, inverse
nic-hdl WK1-AP, inverse
mnt-by MNT-KRNIC-AP, inverse
changed hostmaster@nic.or.kr 20000927
source APNIC
```

Reference: www.apnic.net

209.67.50.203 **futuresite.register.com**

Exodus Communications Inc. (NETBLK-ECI-5)
1605 Wyatt Dr.
Santa Clara, CA 95054
US

Netname: ECI-5
Netblock: 209.67.0.0 - 209.67.255.255
Maintainer: ECI

Coordinator:
Center, Network Control (NOC44-ARIN) CompServ@Exodus.net
(888) 239-6387 (FAX) (888) 239-6387

Domain System inverse mapping provided by:

NS.EXODUS.NET 206.79.230.10
NS2.EXODUS.NET 207.82.198.150

*Rwhois reassignment information for this block is available at:
* rwhois.exodus.net 4321

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

Record last updated on 27-Oct-1998.
Database last updated on 3-Apr-2001 22:30:39 EDT.

Reference: www.arin.net

```
195.56.182.206                                          (unresolved)
inetnum: 195.56.0.0 - 195.56.255.255
netname: HU-DATANET-960426
descr: ALLOCATED BLOCK
descr: Provider Local Registry
descr: DataNet Telecommunication Ltd.
country: HU
admin-c: CN6-RIPE
tech-c: ZR1-RIPE
status: ALLOCATED PA
mnt-by: RIPE-NCC-HM-MNT
changed: hostmaster@ripe.net 19960426
source: RIPE

person: Csaba Nagy
address: DataNet Telecommunications Ltd.
address: Zsigmond ter 10.
address: H-1023 Budapest
address: Hungary
phone: +36 1 3458888
fax-no: +36 1 3458811
e-mail: szncs@datanet.hu
nic-hdl: CN6-RIPE
remarks: abuse@datanet.hu
changed: rzsolt@datanet.hu 19981030
source: RIPE

person: Zsolt Raksanyi
address: GTS-DataNet Telecommunications Ltd.
address: Vaci ut 37/a.
address: H-1134 Budapest
address: Hungary
phone: +36 1 4524451
fax-no: +36 1 4524499
e-mail: rzsolt@datanet.hu
nic-hdl: ZR1-RIPE
remarks: abuse@datanet.hu
changed: rzsolt@datanet.hu 20000225
source: RIPE

                        Reference:  www.ripe.net
```

Of paricular concern here are the heavily targeting hosts that originate from foreign countries such as Israel and Korea; Israel and Korea are often (but not necessarily the only) countries of origin for malicious network traffic.

Watchlisted Hosts

Watchlisted hosts are those that have been identified as malicious, and the Snort sensor has alerted separately to them. The Snort sensor generated 108,319 alerts for watchlisted hosts. The five most prevalent watchlisted hosts are shown below:

Host	Watchlist Alerts	% Watchlist Alerts
212.179.79.2	48786	45.0%
212.179.27.111	39015	36.0%
212.179.95.5	4563	4.2%
212.179.77.20	2353	2.2%
212.179.44.105	1517	1.4%

It is interesting to note the correlation between the top two watchlisted hosts (212.179.79.2 and 212.179.27.111) and the most active targeting hosts from the previous section. Because these are known malicious hosts, and are generating high percentages of traffic, rules should be placed at all ingress chokepoints to deny all traffic originating from these hosts.

Generated Alerts

The following are all of the unique alerts generated by your Snort sensor:

Alert Name	Occurences
Watchlist 000220 IL-ISDNNET-990517	105918
SYN-FIN scan!	51192
DNS udp DoS attack described on unisog	16146
Tiny Fragments - Possible Hostile Activity	5340
connect to 515 from outside	4238
Watchlist 000222 NET-NCFC	2401
WinGate 1080 Attempt	2239
Attempted Sun RPC high port access	2053
Null scan!	826
Queso fingerprint	710
SNMP public access	591
NMAP TCP ping!	558
Russia Dynamo - SANS Flash 28-jul-00	546
SMB Name Wildcard	515
SUNRPC highport access!	204
connect to 515 from inside	159
Broadcast Ping to subnet 70	154
TCP SMTP Source Port traffic	100
Back Orifice	77
External RPC call	59
Probable NMAP fingerprint attempt	8
site exec - Possible wu-ftpd exploit - GIAC000623	3
Happy 99 Virus	1
STATDX UDP attack	1

The five most commonly occuring alerts (Watchlist alerts omitted) are explained below:

SYN-FIN Scan

The SYN-FIN Scan takes advantage of an undefined TCP flag combination in the TCP standard. A SYN flag is used in the initial threeway handshake to start a TCP connection, and the FIN flag is used to gracefully tear down an existing TCP connection. The SYN and FIN flags are direct opposites of each other, and thus

should never be used together. However because their combination is not defined in the TCP standard, the manner in which to handle such anomalous packets was left to the software developers implementing TCP. Therefore, different TCP implementations react differently to this type of malformed header, causing this flag combination to be a fairly effective attack for OS fingerprinting.

References:

- http://www.sans.org/y2k/practical/Markus_DeShon.html
- No relevant CVE reference

DNS UDP DoS Attack Described on Unisog

Not much information could be gathered about this attack, even with a search of the Unisog mailing list archives. The rules database available online for Snort also did not show the rule or one similar to it; it is my guess that this is a custom rule, and not a very well named one at that. (There should be some reference information in the name to correlate this alert to other attacks.) Has this attack been submitted to the CVE yet?

References:

- http://www.theorygroup.com/Archive/Unisog/2001/thrd2.html
- No relevant CVE reference

Tiny Fragments

Packet fragmentation is a normal part of passing IP packets across networks of varying technologies, speeds, and maximum transmission units (MTU). However, fragmentation can be used maliciously by attackers. Because it is the responsibility of the target host to reassemble fragmented packets, it can be vulnerable to anomalous fragmentation. Known fragmentation attacks include a DoS caused by the attacker never sending the last fragment in a fragment chain (causing the victim host to wait until a specified timeout), misaligned or missing fragments in a chain, or a reassembly buffer overflow. Another well-known fragmentation is the "Ping o' Death" attack. Tiny fragments do not indicate the presence of an attack, only the possibility of one.

References:

- www.packetstorm.securify.com/DoS
- CVE-1999-0052
- CVE-1999-0918
- CVE-2000-0305
- CAN-1999-0602

(cve.mitre.org)

Connect to 515 from Outside

Port 515 is the listening port for the lpd (line printer daemon) system on UNIX. It is used for both local and remote printing on UNIX systems. There are however, several exploits with lpd, such that outside connections should not be allowed. (See references below.)

References:

- www.securityfocus.com/templates/archive.pike?list=1&mid=149954
- CVE-1999-0299
- CAN-2000-0839

(cve.mitre.org)

WinGate 1080 Attempt

WinGate is a Windows-based Internet sharing/proxy server solution. It acts as a firewall and supports NAT, Proxy, and LSP. However, there are several security vulnerabilities associated with WinGate. (See references below.)

References:

- www.securityfocus.com (Bugtraq, too many results to list here)
- CVE-1999-0290
- CVE-1999-0291
- CVE-1999-0441
- CVE-1999-0494
- CAN-2000-1048

(cve.mitre.org)

An additional warning: Back Orifice was detected by your Snort sensor! This is sure sign that at least one host, if not more, have been compromised on your network. The offending hosts need to be taken offline immediately, the pertinent data backed up and the operating system and softwares reinstalled and brought up to current patch levels.

Conclusion

During the course of this analysis, several network vulnerabilities were uncovered and compromised hosts discovered. By following the recommendations stated above for the differenent areas of concern, you can remedy some of the current vulnerabilities and operate at a decreased level of risk.

Author's Note: Analysis was performed using both Unix shell functions and Excel for parsing, sorting, and tallying. Excel, however, has a serious shortcoming when working with the amount of data presented here; it can only handle 65536 rows of data.