Global Information Assurance Certification Paper

GIAC Intrusion Detection
Level II Practical Assignment Version 2.7/2.7a
SANS New Orleans
January 27 – February 02, 2001

Nelson Carter

Assignment 1- Network Detects (40 Points)

Detect #1

[**] IDS171/ping zeros [**]
03/13-08:55:47.560468 216.53.156.66 -> x.x.x.2
ICMP TTL:113 TOS:0x0 ID:36897 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:513 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:47.564338 216.53.156.66 -> x.x.x.3
ICMP TTL:113 TOS:0x0 ID:37153 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:769 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:47.566559 216.53.156.66 -> x.x.x.4
ICMP TTL:113 TOS:0x0 ID:37409 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:1025 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:47.569938 216.53.156.66 -> x.x.x.5
ICMP TTL:113 TOS:0x0 ID:37665 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:1281 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:48.549228 216.53.156.66 -> x.x.x.7
ICMP TTL:113 TOS:0x0 ID:44577 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:1793 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:48.562190 216.53.156.66 -> x.x.x.11
ICMP TTL:113 TOS:0x0 ID:45601 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:2817 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:48.565508 216.53.156.66 -> x.x.x.12
ICMP TTL:113 TOS:0x0 ID:45857 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:3073 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:48.643603 216.53.156.66 -> x.x.x.15
ICMP TTL:113 TOS:0x0 ID:46625 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089 Seq:3841 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:49.558135 216.53.156.66 -> x.x.x.18
ICMP TTL:113 TOS:0x0 ID:49185 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:4609 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:50.564051 216.53.156.66 -> x.x.x.30
ICMP TTL:113 TOS:0x0 ID:54305 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:7681 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:53.575100 216.53.156.66 -> x.x.x.61
ICMP TTL:113 TOS:0x0 ID:5154 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:15617 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:54.568690 216.53.156.66 -> x.x.x.66
ICMP TTL:113 TOS:0x0 ID:12322 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:16897 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:54.577986 216.53.156.66 -> x.x.x.69
ICMP TTL:113 TOS:0x0 ID:13090 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:17665 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:55.584839 216.53.156.66 -> x.x.x.80
ICMP TTL:113 TOS:0x0 ID:16674 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:20481 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:57.591782 216.53.156.66 -> x.x.x.99
ICMP TTL:113 TOS:0x0 ID:21794 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:25345 ECHO

[**] IDS171/ping zeros [**]
03/13-08:55:57.597377 216.53.156.66 -> x.x.x.100
ICMP TTL:113 TOS:0x0 ID:22050 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:25601 ECHO

[**] IDS171/ping zeros [**]
03/13-08:56:00.598513 216.53.156.66 -> x.x.x.127
ICMP TTL:113 TOS:0x0 ID:34082 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:32513 ECHO

[**] IDS171/ping zeros [**]
03/13-08:56:10.634480 216.53.156.66 -> x.x.x.230
ICMP TTL:113 TOS:0x0 ID:22819 IpLen:20 DgmLen:60
Type:8 Code:0 ID:25089   Seq:58881 ECHO

Payload Decode example:

Payload length = 1472

```
000 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
010 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
020 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
030 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
040 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
050 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
060 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
070 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
080 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
090 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0a0 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0b0 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0c0 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ..............…
ETC…..
```

1. Source of Trace.
This detect was collected from the network of the company I work for.

2. Detect was generated by:
This detect was generated by the Snort intrusion detection system running on a windows NT server placed on the public side of our firewall.

The format for these log files is as follows:

```
[**] Alert identification messages [**]
Timestamp      Source IP : Source port -> Dest. IP : Dest. port
Protocol    Time toLive     Type of service   Identification #    IP & Datagram length
ICMP- Type #   Code #   ID: #   Seq  #    Type name
```

3. Probability the source address was spoofed:
Depending on how ICMP is used to attack a particular system, spoofing can be very high (in the case of DDoS attacks where the attacker needs no response) or low (in the case of network mapping where the attacker needs a response). The probability that this source address is spoofed is low for two main reasons, the number of attacks is particularly low and the Dest. IP changes with each packet (a clear case of network mapping).

4. Description of attack:
ICMP ping's can be used in many malicious ways, it is used to search the target network for alive hosts in this particular detect. This type of reconnaissance can be used to map out a network for use in future attacks since the attacker has a good idea what hosts are active. This type of attack is currently under review by CVE as CAN-1999-0523

5. Attack mechanism:
The attacker pings the target host(s) in hopes to get a response from the targeted system which will tell the attacker that that host is indeed alive and functioning on the network.

Any “host unreachable “ or “request timed out” messages can be evidence to the attacker that no host is alive there or they are not allowing ICMP traffic.

6. Correlations:
Evidence and information of this type of attack can be found at:

http://whitehats.com/info/IDS171

And

http://www.sans.org/y2k/022101-1300.htm

```
[**] IDS171/ping zeros [**]
02/20-16:11:08.903363 200.52.103.160 -> a.b.20.2
ICMP TTL:50 TOS:0x0 ID:25814
ID:2937   Seq:1  ECHO
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00                                            ..
```

.

7. Evidence of active targeting:
Yes, due to the nature of hitting multiple (not to mention ALL of our public) hosts, I believe this is evidence of a pre attack probe.

8. Severity:

Target Criticality: **5**, Multiple systems where hit including firewall, e-mail and critical Web servers.

Attack Lethality: **2**, for this particular attack it seems to be only reconnaissance.

System Countermeasures: **5,** all machines are up to date with security patches

Network Countermeasures: **5,** Firewall is up to date and the only way in or out

(5 + 2) – (5 + 5) = -3

9. Defensive recommendation:

After reviewing firewall logs this attack was dropped from entering our network

10. Multiple choice test question:

What is the number code for ECHO type in ICMP

a. 4
b. 2
c. 8 =correct
d. 6

Detect #2

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.802609 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:56 IpLen:20 DgmLen:1064 DF
A* Seq: 0x62248B5 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.817792 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:824 IpLen:20 DgmLen:1064 DF
A* Seq: 0x6224CB5 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.819140 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:1080 IpLen:20 DgmLen:374 DF
AP Seq: 0x62250B5 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.855384 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:1592 IpLen:20 DgmLen:1064 DF
A* Seq: 0x6225203 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.866700 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:1848 IpLen:20 DgmLen:1064 DF
A* Seq: 0x6225603 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.879010 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:2104 IpLen:20 DgmLen:1064 DF
A* Seq: 0x6225A03 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.890422 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:2360 IpLen:20 DgmLen:1064 DF
A* Seq: 0x6225E03 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/13-14:35:52.893380 63.122.201.140:443 -> x.x.x.230:20432
TCP TTL:115 TOS:0x0 ID:2616 IpLen:20 DgmLen:632 DF
AP Seq: 0x6226203 Ack: 0x1F0AFC Win: 0x1D59 TcpLen: 20

Payload Decode Example:

```
length = 139

000 : 48 54 54 50 2F 31 2E 31 20 33 30 34 20 4E 6F 74   HTTP/1.1 304 Not
010 : 20 4D 6F 64 69 66 69 65 64 0D 0A 53 65 72 76 65    Modified..Serve
020 : 72 3A 20 4D 69 63 72 6F 73 6F 66 74 2D 49 49 53   r: Microsoft-IIS
030 : 2F 34 2E 30 0D 0A 44 61 74 65 3A 20 54 75 65 2C   /4.0..Date: Tue,
040 : 20 30 36 20 4D 61 72 20 32 30 30 31 20 31 34 3A    06 Mar 2001 14:
050 : 32 32 3A 30 39 20 47 4D 54 0D 0A 45 54 61 67 3A   22:09 GMT..ETag:
060 : 20 22 30 32 61 62 37 63 36 66 36 33 63 30 31 3A    "02ab7c6f63c01:
070 : 64 30 34 22 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65   d04"..Content-Le
080 : 6E 67 74 68 3A 20 30 0D 0A 0D 0A                  ngth: 0....
```

1. Source of Trace.
This detect was collected from the network of the company I work for.

2. Detect was generated by:
This detect was generated by the Snort intrusion detection system running on a windows NT server placed on the public side of our firewall.

The format for these log files is as follows:

[**] Alert identification messages [**]
Timestamp Source IP : Source port -> Dest. IP : Dest. port
Protocol Time toLive Type of service Identification # IP & Datagram length
Flags Sequece # Ackowldgement # Window size TCP length

3. Probability the source address was spoofed:

Low, the client in this case needs response from the handler . The client , I believe, is a borrowed IP from an ISP

Netname: UUNET63
Netblock: 63.64.0.0 - 63.127.255.255
Maintainer: UU

Coordinator:
UUNET, AlterNet - Technical Support (OA12-ARIN) help@UUNET.UU.NET
800-900-0241.

4. Description of attack:

This is an example of a possible communication from a Shaft client to a handler in an attempt to send commands to agents that will in turn execute an attack on a pre-determined host. This type of attack is currently under review by CVE as:
CAN-2000-0138

5. Attack mechanism:

The attacker communicates commands to the handler to source port 20432 to do a number of things, check what agents are controlled from that host, forward commands to agents and to start the attack on the victim(s).

6. Correlations:

Evidence and information of this type of attack can be found at:

http://whitehats.com/info/IDS254

and :

Other examples of this detect from the same network:

[**] IDS254/ddos-shaft-client-to-handler [**]
03/06-09:23:06.010722 199.92.202.10:80 -> x.x.x.230:20432
TCP TTL:112 TOS:0x0 ID:18196 IpLen:20 DgmLen:179 DF
AP Seq: 0x16BF6E0D Ack: 0x1AF98 Win: 0x2238 TcpLen: 20

[**] IDS254/ddos-shaft-client-to-handler [**]
03/09-13:53:30.122583 207.137.169.20:80 -> x.x.x.230:20432
TCP TTL:53 TOS:0x0 ID:49712 IpLen:20 DgmLen:692 DF
***AP**F Seq: 0x2AFB0F23 Ack: 0x14A75B0 Win: 0x7D78 TcpLen: 20

Also from: http://www.sans.org/y2k/shaft.htm

```
Sun 11/28 21:39:22 tcp 129.22.64.17.53982 <-> x.x.x.x.21
Sun 11/28 21:39:56 udp x.x.x.x.33198 -> 129.22.64.17.20433
Sun 11/28 21:45:20 udp 129.22.64.17.1765 -> x.x.x.x.18753
Sun 11/28 21:45:20 udp x.x.x.x.33199 -> 129.22.64.17.20433
Sun 11/28 21:45:59 udp 129.22.64.17.1866 -> x.x.x.x.18753
Sun 11/28 21:45:59 udp x.x.x.x.33200 -> 129.22.64.17.20433
Sun 11/28 21:45:59 udp 129.22.64.17.1968 -> x.x.x.x.18753
Sun 11/28 21:45:59 udp 129.22.64.17.1046 -> x.x.x.x.18753
Sun 11/28 21:45:59 udp 129.22.64.17.1147 -> x.x.x.x.18753
Sun 11/28 21:45:59 udp 129.22.64.17.1248 -> x.x.x.x.18753
Sun 11/28 21:45:59 udp 129.22.64.17.1451 -> x.x.x.x.18753
Sun 11/28 21:46:00 udp x.x.x.x.33201 -> 129.22.64.17.20433
Sun 11/28 21:46:00 udp x.x.x.x.33202 -> 129.22.64.17.20433
Sun 11/28 21:46:01 udp x.x.x.x.33203 -> 129.22.64.17.20433
Sun 11/28 21:48:37 udp 129.22.64.17.1037 -> x.x.x.x.18753
Sun 11/28 21:48:37 udp 129.22.64.17.1239 -> x.x.x.x.18753
Sun 11/28 21:48:37 udp 129.22.64.17.1340 -> x.x.x.x.18753
Sun 11/28 21:48:37 udp 129.22.64.17.1442 -> x.x.x.x.18753
Sun 11/28 21:48:38 udp x.x.x.x.33204 -> 129.22.64.17.20433
Sun 11/28 21:48:38 udp x.x.x.x.33205 -> 129.22.64.17.20433
Sun 11/28 21:48:38 udp x.x.x.x.33206 -> 129.22.64.17.20433
Sun 11/28 21:48:56 udp 129.22.64.17.1644 -> x.x.x.x.18753
Sun 11/28 21:48:56 udp x.x.x.x.33207 -> 129.22.64.17.20433
```

```
Sun 11/28 21:49:59 udp x.x.x.x.33208 -> 129.22.64.17.20433
Sun 11/28 21:50:00 udp x.x.x.x.33209 -> 129.22.64.17.20433
Sun 11/28 21:50:14 udp 129.22.64.17.1747 -> x.x.x.x.18753
Sun 11/28 21:50:14 udp x.x.x.x.33210 -> 129.22.64.17.20433
```

7. Evidence of active targeting:
In the case this particular detect and the others I have seen on my network, this is NOT enough evidence of active targeting, if fact, I believe this is a false positive generated by snort. All of the source ports detected are either 80 or 443 which leads me to believe that this is return traffic to a personal ISP web page or returning information request to one of our systems (**Web Sphere**) which uses ports in the 20,xxx range for certain custom applications. Also there appears to be no Shaft commands in any of the payload decodes.

8. Severity:

Target Criticality: **5**, Multiple systems where hit including firewall, e-mail and critical Web servers.

Attack Lethality: **1**, false positive

System Countermeasures: **5,** all machines are up to date with security patches

Network Countermeasures: **5,** Firewall is up to date and the only way in or out

(5 + 1) – (5 + 5) = -4

9. Defensive recommendation:

Since this appears to be a false positive none is needed, however it would not hurt to scan for this open port and to change the ports used for this Web Sphere app

10. Multiple choice test question:

What is the default port for client to handler shaft attacks

a. 327703
b. 18753
c. 20433
d. 20432 = correct

Detect #3

[**] IDS175/socks-probe [**]
03/09-17:39:03.762401 194.230.227.191:3943 -> x.x.x.3:1080
TCP TTL:107 TOS:0x0 ID:62451 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2B6DF60 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.769418 194.230.227.191:3944 -> x.x.x.4:1080
TCP TTL:107 TOS:0x0 ID:62452 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2B7B16D Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.772934 194.230.227.191:3945 -> x.x.x.5:1080
TCP TTL:107 TOS:0x0 ID:62453 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2B86566 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.779104 194.230.227.191:3947 -> x.x.x.7:1080
TCP TTL:107 TOS:0x0 ID:62455 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2B99FCE Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.803797 194.230.227.191:3951 -> x.x.x.11:1080
TCP TTL:107 TOS:0x0 ID:62459 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2BC4051 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.808544 194.230.227.191:3952 -> x.x.x.12:1080
TCP TTL:107 TOS:0x0 ID:62460 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2BD0F62 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.829782 194.230.227.191:3955 -> x.x.x.15:1080
TCP TTL:107 TOS:0x0 ID:62463 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2BF1023 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:03.846238 194.230.227.191:3959 -> x.x.x.18:1080
TCP TTL:107 TOS:0x0 ID:62467 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE2C25D36 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

[**] IDS175/socks-probe [**]
03/09-17:39:04.091591 194.230.227.191:3919 -> x.x.x.30:1080
TCP TTL:107 TOS:0x0 ID:62475 IpLen:20 DgmLen:48 DF
******S* Seq: 0xE29BFB75 Ack: 0x0 Win: 0x2238 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

1. Source of Trace.
This detect was collected from the network of the company I work for.

2. Detect was generated by:
This detect was generated by the Snort intrusion detection system running on a windows NT server placed on the public side of our firewall.

The format for these log files is as follows:

[**] Alert identification messages [**]
Timestamp Source IP : Source port -> Dest. IP : Dest. port
Protocol Time toLive Type of service Identification # IP & Datagram length
Flags Sequece # Ackowldgement # Window size TCP length
TCP options

3. Probability the source address was spoofed:
The probability that this source was spoofed can be very high in the case of an attacker sending information through the sock port, however in this can it is a reconnaissance probe checking for open socks ports , so I will have to say in this case the probability is low

4. Description of attack:
The attacker in this case is hitting public host on port 1080 to check for open and vulnerable socks ports

5. Attack mechanism:

The attack crafts SYN packets to a specific ip and port, 1080 or SOCKS in this case, in hopes to get a response that this port is open and functioning on a particular host. In this case TCP options are set to determine the OS as well.

6. Correlations:

Evidence and information of this type of attack can be found at:

http://whitehats.com/info/IDS175

And:

from: http://www.sans.org/y2k/030601.htm

```
Mar  1 17:18:00 172.167.243.197:3896 -> xxx.xxx.xxx.216:1080 SYN ******S*
Mar  1 17:18:02 172.167.243.197:3880 -> xxx.xxx.xxx.200:1080 SYN ******S*
Mar  1 17:18:02 172.167.243.197:3877 -> xxx.xxx.xxx.197:1080 SYN ******S*
Mar  1 17:18:02 172.167.243.197:3874 -> xxx.xxx.xxx.194:1080 SYN ******S*
Mar  1 17:18:02 172.167.243.197:3878 -> xxx.xxx.xxx.198:1080 SYN ******S*
Mar  1 17:18:02 172.167.243.197:3875 -> xxx.xxx.xxx.195:1080 SYN ******S*
Mar  1 17:18:02 172.167.243.197:3879 -> xxx.xxx.xxx.1xxx:1080 SYN ******S*
```

7. Evidence of active targeting:

This is evidence of active targeting ,because the attacker is hitting multiple public host in a an effort to find which ,if any, of these hosts are running a socks proxy

8. Severity:

Target Criticality: **5**, Multiple systems where hit including firewall, e-mail and critical Web servers.

Attack Lethality: **2**, for this particular attack it seems to be only reconnaissance.

System Countermeasures: **5,** all machines are up to date with security patches

Network Countermeasures: **5,** Firewall is up to date and the only way in or out

(5 + 2) – (5 + 5) = -3

9. Defensive recommendation:

After reviewing firewall logs this attack was dropped from entering our network

10. Multiple choice test question:

Port 1080 is typically use for

a. secure FTP
b. proxy servers =correct
c. Sub Seven exploit
d. portmapper

Detect #4

INVALID ACK
Feb 19 12:26:39 192.193.195.178:443 -> x.x.x.230:52743 INVALIDACK ***A*R*F
Feb 19 12:26:39 192.193.195.178:443 -> x.x.x.230:52746 INVALIDACK ***A*R*F
Feb 19 12:26:41 192.193.195.178:443 -> x.x.x.230:52744 INVALIDACK ***A*R*F
Feb 19 12:26:41 192.193.195.178:443 -> x.x.x.230:52753 INVALIDACK ***A*R*F
Feb 19 12:27:17 192.193.195.178:443 -> x.x.x.230:52750 INVALIDACK ***A*R*F
Feb 19 12:28:03 192.193.195.178:443 -> x.x.x.230:52747 INVALIDACK ***A*R*F
Feb 19 12:34:58 192.193.195.178:443 -> x.x.x.230:52960 INVALIDACK ***A*R*F
Mar 16 14:28:37 192.193.195.178:443 -> x.x.x.230:45681 INVALIDACK ***A*R*F
Mar 16 14:28:37 192.193.195.178:443 -> x.x.x.230:45682 INVALIDACK ***A*R*F
Mar 16 14:28:38 192.193.195.178:443 -> x.x.x.230:45688 INVALIDACK ***A*R*F
Mar 16 14:28:39 192.193.195.178:443 -> x.x.x.230:45690 INVALIDACK ***A*R*F

1. Source of Trace.
This detect was collected from the network of the company I work for.

2. Detect was generated by:
This detect was generated by the Snort intrusion detection system running on a windows NT server placed on the public side of our firewall.

The format for these log files is as follows:

Timestamp Source IP : Source port -> Dest. IP : Dest. Port Message Flags

3. Probability the source address was spoofed:

The probability in this case is very high because this appears to be an attack via use of a crafted packet used to "confuse" and bring down a selected host.

4. Description of attack:
This attack uses invalid flag combinations in order to get a desired result such as hanging up the targeted host or possibly bringing in down completely.

5. Attack mechanism:
The attacker has to use software to create a packet that has the ACK, RST and FIN flag bits set, and then it is sent to a target in attempt to gain certain results.

6. Correlations:

I have not seen this particular attack before and I have searched quit a bit for ***exact*** coralations with little luck. The ONLY thing that seems to relate are articles explaining the use of load balancing units that may be used on certain web sites. Here are some examples:

http://www.securityfocus.com/archive/75/159701

```
I believe that this is some sort of fallout from
> the load balancing systems.  These RST are from the server farm behind
> the load balancer and represent real responses to sessions initiated on
> your network.  That is why the source port is 80.  I run argus which
> logs all traffic and this is what I see on close examination
>
> time T  localIP:hiportnum -> www.bigname.com:80  - normal session
> time T+(up to 5 minutes) otherIP:80 -> localIP:hiportnum  - RST

I have had a chance to look at the two way conversation involved with
one of these a bit more. It looks like the RST with the odd extra bit
flipped is a response to what the remote HTTP server must be seeing as
a surprise FIN,
```

```
10:21:28.396870 aaa.bbb.cc2.84.38277 > 205.188.144.231.80: S
2530953764:2530953764(0) win 8760 <mss 1460> (DF) (ttl 254, id 4724)
10:21:28.486222 205.188.144.231.80 > aaa.bbb.cc2.84.38277: S
704124420:704124420(0) ack 2530953765 win 8760 <mss 1460> (DF) (ttl 240, id
7755)
10:21:28.487653 aaa.bbb.cc2.84.38277 > 205.188.144.231.80: . 1:1(0) ack 1 win
8760 (DF) (ttl 254, id 4725)
10:21:28.494449 aaa.bbb.cc2.84.38277 > 205.188.144.231.80: P 1:1323(1322) ack 1
win 8760 (DF) (ttl 254, id 4726)
10:21:28.597889 205.188.144.231.80 > aaa.bbb.cc2.84.38277: . 1:1(0) ack 1323
win 33580 (DF) (ttl 240, id 7756)
10:21:28.603793 205.188.144.231.80 > aaa.bbb.cc2.84.38277: P 1:681(680) ack
1323 win 33580 (DF) (ttl 240, id 7757)
10:21:28.654635 aaa.bbb.cc2.84.38277 > 205.188.144.231.80: . 1323:1323(0) ack
681 win 8760 (DF) (ttl 254, id 4727)
10:21:58.597954 205.188.144.231.80 > aaa.bbb.cc2.84.38277: F 681:681(0) ack
1323 win 33580 (DF) (ttl 240, id 7758)
10:21:58.599507 aaa.bbb.cc2.84.38277 > 205.188.144.231.80: . 1323:1323(0) ack
682 win 8760 (DF) (ttl 254, id 4728)
10:51:02.456094 aaa.bbb.cc2.84.38277 > 205.188.144.231.80: F 1323:1323(0) ack
682 win 8760 (DF) (ttl 254, id 40651)
10:51:02.546232 205.188.144.231.80 > aaa.bbb.cc2.84.38277: R [CWR]
704125102:704125102(0) win 0 (DF) (ttl 49, id 24447)
```

AND:

From: http://www.sans.org/y2k/012900.htm

[This is definitely an oddball. PUSH URG RST? I have seen some web servers that will send out packets in this nature, but the source's name reminds me of a firewall. I have seen some load balancing solutions and some web sites generate this pattern. Could be nothing... most likely.]

```
Jan 27 09:14:33.186 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[] Port 0->65535): Bad IP
Header (received on interface x.x.x.x)
Jan 27 09:14:35.523 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[PUSH RST] Port
27960->27960): Restricted Port: Protocol=TCP[PUSH RST] Port 27960->27960
(received on interface x.x.x.x)
Jan 27 09:14:41.490 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[] Port 30720->32824):
Restricted Port: Protocol=TCP[] Port 30720->32824 (received on interface
x.x.x.x)
Jan 27 09:16:51.648 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[PUSH] Port 18248->14383):
Restricted Port: Protocol=TCP[PUSH] Port 18248->14383 (received on
interface x.x.x.x)
Jan 27 09:18:23.595 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[] Port 30720->32784):
Restricted Port: Protocol=TCP[] Port 30720->32784 (received on interface
x.x.x.x)
Jan 27 09:20:33.625 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[] Port 0->65535): Bad IP
Header (received on interface x.x.x.x)
Jan 27 09:21:36.097 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[FIN] Port 1034->53):
Restricted Port: Protocol=TCP[FIN] Port 1034->53 (received on interface
```

```
x.x.x.x)
Jan 27 09:21:45.044 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[] Port 0->65535): Bad IP
Header (received on interface x.x.x.x)
Jan 27 09:23:11.566 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[FIN] Port 30721->32800):
Restricted Port: Protocol=TCP[FIN] Port 30721->32800 (received on interface
x.x.x.x)
Jan 27 09:23:23.071 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[] Port 0->65535): Bad IP
Header (received on interface x.x.x.x)
Jan 27 09:23:31.446 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[SYN PUSH URG FIN RST] Port
25455->27904): Restricted Port: Protocol=TCP[SYN PUSH URG FIN RST] Port
25455->27904 (received on interface x.x.x.x)
Jan 27 09:23:35.273 OurFW kernel: 347 Possible Port Scan detected on
Interface x.x.x.x (194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[ACK]
Port 61841->119)
Jan 27 09:25:57.684 OurFW kernel: 226 IP packet dropped
(194.217.188.35->OurFW[x.x.x.x]: Protocol=TCP[SYN FIN RST] Port
20039->20295): Restricted Port: Protocol=TCP[SYN FIN RST] Port 20039->20295
(received on interface x.x.x.x)
```

OR: (Bugtrac search on RST-FIN)

From: http://www.securityfocus.com/archive/1/20256

```
I have seen this also in a Floodgate-1 machine that was positioned outside
the firewall.  Flodgate-1 is Checkpoint's bandwidth management solution
which presumably uses the same state engine.

In this particular instance the firewall that had been deployed was not
capabale of running Floodgate on the same machine so Floodgate had been
deployed on a relatively sacrificial host that was positioned between the
firewall and the Internet router.  As floodgate doesn't do any traffic
filtering, when I portscanned it from an external point the connections
were allowed through to the firewall, where they were dropped without a
NACK/RST/FIN coming back the other way.  The machine consistently died
after a matter of minutes.

Some more graceful error handling on Checkpoint's behalf would probably be
nice.

Regards,
Dave Taylor
```

7. Evidence of active targeting:

It appears to me that this IS evidence of active targeting , however it could also be soothing similar to the above mentioned or just plain faulty hardware/software.

8. Severity:

Since I am not sure here, I will assume the worst

Target Criticality: **5**, This could be a direct attack

Attack Lethality: **4**, If this is an attack on a FW1 vulnerability it could potentially bring down the firewall

System Countermeasures: **5,** All machines are up to date with security patches

Network Countermeasures: **2,** Since this could be a direct attack on FW1

(5 +4) – (5 + 2) = 2

9. Defensive recommendation:
Research and review all potential vulnerabilities of FW1 and update to latest patches and write additional ACLs for the perimeter routers to prevent this

10. Multiple choice test question:

Which RFC is responsible for defining normal TCP functions
a. RFC 793 = correct
b. RFC 768
c. RFC 791
d. RFC 792

Detect #5

```
# STRANGE ALERT THAT GOES TO A DUTCH SITE
[**] spp_http_decode: IIS Unicode attack detected [**]
02/28-08:49:52.082963 x.x.x.230:20902 -> 194.25.242.201:80
TCP TTL:127 TOS:0x0 ID:10762 IpLen:20 DgmLen:1141 DF
***AP*** Seq: 0xDB065  Ack: 0x2EBF31F5  Win: 0x2238  TcpLen: 20
GET /ex/shak/index.html HTTP/1.0..Referer: http://www.netscape.c
om/..Connection: Keep-Alive..User-Agent: Mozilla/4.61 [en] (WinN
T; I)..Host: home.netscape.com..Accept: image/gif, image/x-xbitm
ap, image/jpeg, image/pjpeg, image/png, */*..Accept-Encoding: gz
ip..Accept-Language: en..Accept-Charset: iso-8859-1,*,utf-8..Coo
kie: UIDC=199.93.167.1:0946412322:933554; msInterfaceID=5; SITES
ERVER=ID=3cc3a03893b3b9b16c0169e27242130f; NS_REG2_USERLOGIN=SHA
1=.Y....a...y02%18a%D1%CB%E4y%C0%3C%A7D%17%40%99na[-]UR2%5FUSER%
5FID=khua27[-]UR2%5FLOGGED%5FIN=EXPRESS; NSCP_USER_LOGIN1_NEW=SH
A1=%EC%2C%13%AB1%ED9%A8%E2%84%D0%D4%01%C01%A0%A3%DF%21%C2[-]UR2%
5FUSER%5FID=khua27[-]UR2%5FOLD%5FUSER%5FID=khua27[-]UR2%5FLOGGED
%5FIN=EXPRESS; NSCP_USER_LOGIN1=SHA1=%EC%2C%13%AB1%ED9%A8%E2%84%
D0%D4%01%C01%A0%A3%DF%21%C2[-]UR2%5FUSER%5FID=khua27[-]UR2%5FOLD
%5FUSER%5FID=khua27[-]UR2%5FLOGGED%5FIN=EXPRESS; NS_REG=SHA1=%C9
sNI%B0%FCp%C5y%FA%07%8FP%17%BFZ%E9XYH[-]UR%5FEMAIL=khua27%40nets
cape%2Enet[-]UR%5FREG%5FID=38087666%3AWEBv1; HITO_VISITS=A417030
21+27F440*E7725*2+34B642*E7FC5*7; NS_WM=khua27:0:200112150230[-]
WM_LOGGED....
```

=+

[**] spp_http_decode: IIS Unicode attack detected [**]
02/28-08:49:52.082963 x.x.x.230:20902 -> 194.25.242.201:80
TCP TTL:127 TOS:0x0 ID:10762 IpLen:20 DgmLen:1141 DF
AP Seq: 0xDB065 Ack: 0x2EBF31F5 Win: 0x2238 TcpLen: 20
GET /ex/shak/index.html HTTP/1.0..Referer: http://www.netscape.c
om/..Connection: Keep-Alive..User-Agent: Mozilla/4.61 [en] (WinN
T; I)..Host: home.netscape.com..Accept: image/gif, image/x-xbitm
ap, image/jpeg, image/pjpeg, image/png, */*..Accept-Encoding: gz
ip..Accept-Language: en..Accept-Charset: iso-8859-1,*,utf-8..Coo
kie: UIDC=199.93.167.1:0946412322:933554; msInterfaceID=5; SITES
ERVER=ID=3cc3a03893b3b9b16c0169e27242130f; NS_REG2_USERLOGIN=SHA
1=.Y....a...y02%18a%D1%CB%E4y%C0%3C%A7D%17%40%99na[-]UR2%5FUSER%
5FID=khua27[-]UR2%5FLOGGED%5FIN=EXPRESS; NSCP_USER_LOGIN1_NEW=SH
A1=%EC%2C%13%AB1%ED9%A8%E2%84%D0%D4%01%C01%A0%A3%DF%21%C2[-]UR2%
5FUSER%5FID=khua27[-]UR2%5FOLD%5FUSER%5FID=khua27[-]UR2%5FLOGGED
%5FIN=EXPRESS; NSCP_USER_LOGIN1=SHA1=%EC%2C%13%AB1%ED9%A8%E2%84%
D0%D4%01%C01%A0%A3%DF%21%C2[-]UR2%5FUSER%5FID=khua27[-]UR2%5FOLD
%5FUSER%5FID=khua27[-]UR2%5FLOGGED%5FIN=EXPRESS; NS_REG=SHA1=%C9
sNI%B0%FCp%C5y%FA%07%8FP%17%BFZ%E9XYH[-]UR%5FEMAIL=khua27%40nets
cape%2Enet[-]UR%5FREG%5FID=38087666%3AWEBv1; HITO_VISITS=A417030
21+27F440*E7725*2+34B642*E7FC5*7; NS_WM=khua27:0:200112150230[-]
WM_LOGGED....

At First this seemed like a Snort pre processor false positive, but Then the Dutch traffic returns fast and loud (via custom filter I wrote)

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.503645 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:241 TOS:0x0 ID:7280 IpLen:20 DgmLen:914 DF
A* Seq: 0xEB0860E0 Ack: 0x1B83009 Win: 0x832C TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.512086 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:241 TOS:0x0 ID:7281 IpLen:20 DgmLen:1500 DF
A* Seq: 0xEB08644A Ack: 0x1B83009 Win: 0x832C TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.512717 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:255 TOS:0x0 ID:0 IpLen:20 DgmLen:2380
AP Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.525933 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:241 TOS:0x0 ID:7282 IpLen:20 DgmLen:1500 DF
A* Seq: 0xEB0869FE Ack: 0x1B83009 Win: 0x832C TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.526614 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:255 TOS:0x0 ID:0 IpLen:20 DgmLen:1517
AP Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.536547 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:241 TOS:0x0 ID:7283 IpLen:20 DgmLen:1500 DF
A* Seq: 0xEB086FB2 Ack: 0x1B83009 Win: 0x832C TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.544449 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:241 TOS:0x0 ID:7284 IpLen:20 DgmLen:1500 DF
A* Seq: 0xEB087566 Ack: 0x1B83009 Win: 0x832C TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.545106 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:255 TOS:0x0 ID:0 IpLen:20 DgmLen:2941
AP Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.552291 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:241 TOS:0x0 ID:7285 IpLen:20 DgmLen:1500 DF
A* Seq: 0xEB087B1A Ack: 0x1B83009 Win: 0x832C TcpLen: 20

[**] DUTCH TRAFFIC IN BOUND [**]
02/28-15:42:33.552986 194.25.242.201:80 -> x.x.x.230:35556
TCP TTL:255 TOS:0x0 ID:0 IpLen:20 DgmLen:1538
AP Seq: 0x0 Ack: 0x0 Win: 0x0 TcpLen: 20

here is a decode example of these packets:

```
length = 267

000 : 48 54 54 50 2F 31 2E 31 20 32 30 30 20 4F 4B 0D   HTTP/1.1 200 OK.
010 : 0A 53 65 72 76 65 72 3A 20 4E 65 74 73 63 61 70   .Server: Netscap
020 : 65 2D 45 6E 74 65 72 70 72 69 73 65 2F 34 2E 31   e-Enterprise/4.1
030 : 0D 0A 44 61 74 65 3A 20 54 75 65 2C 20 30 36 20   ..Date: Tue, 06
040 : 4D 61 72 20 32 30 30 31 20 32 31 3A 31 35 3A 32   Mar 2001 21:15:2
050 : 33 20 47 4D 54 0D 0A 53 65 74 2D 43 6F 6F 6B 69   3 GMT..Set-Cooki
060 : 65 3A 20 55 49 44 43 3D 31 32 2E 34 2E 33 2E 32   e: UIDC=x.x.x.2
070 : 33 30 3A 30 39 38 33 39 31 33 33 32 33 3A 37 31   30:0983913323:71
080 : 34 38 32 36 3B 64 6F 6D 61 69 6E 3D 2E 6E 65 74   4826;domain=.net
090 : 73 63 61 70 65 2E 63 6F 6D 3B 70 61 74 68 3D 2F   scape.com;path=/
0a0 : 3B 65 78 70 69 72 65 73 3D 33 31 2D 44 65 63 2D   ;expires=31-Dec-
0b0 : 32 30 31 30 20 32 33 3A 35 39 3A 35 39 20 47 4D   2010 23:59:59 GM
0c0 : 54 0D 0A 43 6F 6E 74 65 6E 74 2D 74 79 70 65 3A   T..Content-type:
0d0 : 20 74 65 78 74 2F 68 74 6D 6C 0D 0A 43 6F 6E 74    text/html..Cont
0e0 : 65 6E 74 2D 6C 65 6E 67 74 68 3A 20 32 34 39 0D   ent-length: 249.
0f0 : 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 6B 65 65   .Connection: kee
100 : 70 2D 61 6C 69 76 65 0D 0A 0D 0A                  p-alive....
```

```
length = 249

000 : 3C 48 54 4D 4C 3E 0A 3C 48 45 41 44 3E 0A 3C 53   <HTML>.<HEAD>.<S
010 : 43 52 49 50 54 20 4C 41 4E 47 55 41 47 45 3D 22   CRIPT LANGUAGE="
020 : 4A 61 76 61 53 63 72 69 70 74 31 2E 31 22 3E 0A   JavaScript1.1">.
030 : 3C 21 2D 2D 0A 64 6F 63 75 6D 65 6E 74 2E 6C 6F   <!--.document.lo
040 : 63 61 74 69 6F 6E 2E 72 65 70 6C 61 63 65 28 22   cation.replace("
050 : 68 74 74 70 3A 2F 2F 68 6F 6D 65 2E 6E 65 74 73   http://home.nets
060 : 63 61 70 65 2E 63 6F 6D 2F 65 78 2F 73 68 61 6B   cape.com/ex/shak
070 : 2F 69 6E 64 65 78 2E 68 74 6D 6C 22 29 3B 0A 2F   /index.html");./
080 : 2F 2D 2D 3E 0A 3C 2F 53 43 52 49 50 54 3E 0A 3C   /-->.</SCRIPT>.<
090 : 4D 45 54 41 20 48 54 54 50 2D 45 51 55 49 56 3D   META HTTP-EQUIV=
0a0 : 22 52 65 66 72 65 73 68 22 20 43 4F 4E 54 45 4E   "Refresh" CONTEN
0b0 : 54 3D 22 30 3B 20 55 52 4C 3D 68 74 74 70 3A 2F   T="0; URL=http:/
0c0 : 2F 68 6F 6D 65 2E 6E 65 74 73 63 61 70 65 2E 63   /home.netscape.c
0d0 : 6F 6D 2F 65 78 2F 73 68 61 6B 2F 69 6E 64 65 78   om/ex/shak/index
0e0 : 2E 68 74 6D 6C 22 3E 0A 3C 2F 48 45 41 44 3E 0A   .html">.</HEAD>.
0f0 : 3C 2F 48 54 4D 4C 3E 0A 0A                        </HTML>..
```

This traffic came in at a frequency of about 250 packets per minute

1. Source of Trace.
This detect was collected from the network of the company I work for.

2. Detect was generated by:
This detect was generated by the Snort intrusion detection system running on a windows NT server placed on the public side of our firewall.

The format for these log files is as follows:

[**] Alert identification messages [**]
Timestamp Source IP : Source port -> Dest. IP : Dest. port
Protocol Time toLive Type of service Identification # IP & Datagram length
Flags Sequece # Ackowldgement # Window size TCP length
Payload Decode

3. Probability the source address was spoofed:
I believe this traffic was spoofed for two main reasons:

First the IP is part of a German ISP and I got no response from the ISP concerning this particular ip. Here is the info:

inetnum: 194.25.242.0 - 194.25.243.255
netname: DTAG-ONLINE2
descr: Deutsche Telekom AG
country: DE
admin-c: RH2086-RIPE
tech-c: AH12705-RIPE
tech-c: ST5359-RIPE
status: ASSIGNED PA
remarks: **********

```
remarks:    * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS, *
remarks:    * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.   *
remarks:    ************************************************************
notify:     auftrag@nic.telekom.de
notify:     dbd@nic.dtag.de
mnt-by:      DTAG-NIC
changed:     auftrag@nic.telekom.de 20010321
source:     RIPE

route:      194.25.0.0/16
descr:      Deutsche Telekom AG, Internet service provider
origin:     AS3320
mnt-by:      DTAG-RR
changed:     bp@NIC.DTAG.DE 19960215
changed:     bp@NIC.DTAG.DE 19980423
source:     RIPE
```

AND

According to the decode the incoming traffic was re-directing Netscape traffic to this specific IP. I contacted AOL/Netscape and they DO NOT have a server at that particular IP address.

4. Description of attack:

To be honest I am a little thrown by this because the packets them selves don't look out of the ordinary, but the 194.25.242.201 IP does not seem to belong to anyone so these are my thoughts so far.

This attack could be a result of a number of different things, but to me it almost seems to be some sort of DDoS just for the simple fact that there are SO many packets coming in. As a result the packet going out from my network (which seem to be normal, but in the big picture could be some kind of Trojan) a large # of redirects seem to be going back to netscape.com.

5. Attack mechanism:

Since I only started to use Snort at the beginning of February, I was unable to catch the very beginning of these packets. I believe this started as a link (possibly through e-mail) to the attackers spoofed site, then the user on our network seems to push cookies to that site. After that it seem that a large flood of these packet come back and set a cookie for Netscape then run some kind of script to redirect them back.
Also I have noticed that some of our clients have been complaining that they cannot get to any Netscape home pages. Upon looking further into this I noticed that there has been a change in a DNS record somewhere that redirects netscape.com to this IP address although AOL/Netscape claims it is NOT on of theirs.

6. Correlations:

I have had some E-mail discussions with Max Vision from Whitehats.com about this and he kind of got the same type of response:

```
03/09-01:37:28.131460 194.25.242.201:80 -> maxtest:2270
TCP TTL:238 TOS:0x0 ID:58304 IpLen:20 DgmLen:283 DF
***AP*** Seq: 0x979D25B0  Ack: 0xFB8BDE13  Win: 0x832C  TcpLen: 20
48 54 54 50 2F 31 2E 31 20 32 30 30 20 4F 4B 0D  HTTP/1.1 200 OK.
0A 53 65 72 76 65 72 3A 20 4E 65 74 73 63 61 70  .Server: Netscap
65 2D 45 6E 74 65 72 70 72 69 73 65 2F 34 2E 31  e-Enterprise/4.1
0D 0A 44 61 74 65 3A 20 46 72 69 2C 20 30 39 20  ..Date: Fri, 09
4D 61 72 20 32 30 30 31 20 30 38 3A 33 36 3A 33  Mar 2001 08:36:3
32 20 47 4D 54 0D 0A 53 65 74 2D 43 6F 6F 6B 69  2 GMT..Set-Cooki
65 3A 20 55 49 44 43 3D xx xx 2E xx 2E xx xx 2E  e: UIDC=xx.x.xx.
xx xx 3A 30 39 38 34 31 32 36 39 39 32 3A 39 31  xx:0984126992:91
31 33 34 34 3B 64 6F 6D 61 69 6E 3D 2E 6E 65 74  1344;domain=.net
73 63 61 70 65 2E 63 6F 6D 3B 70 61 74 68 3D 2F  scape.com;path=/
3B 65 78 70 69 72 65 73 3D 33 31 2D 44 65 63 2D  ;expires=31-Dec-
32 30 31 30 20 32 33 3A 35 39 3A 35 39 20 47 4D  2010 23:59:59 GM
54 0D 0A 43 6F 6E 74 65 6E 74 2D 74 79 70 65 3A  T..Content-type:
20 74 65 78 74 2F 68 74 6D 6C 0D 0A 43 6F 6E 74   text/html..Cont
65 6E 74 2D 6C 65 6E 67 74 68 3A 20 32 34 39 0D  ent-length: 249.
0A 0D 0A                                         ...

03/09-01:37:28.132045 194.25.242.201:80 -> maxtest:2270
TCP TTL:238 TOS:0x0 ID:58305 IpLen:20 DgmLen:289 DF
***AP*** Seq: 0x979D26A3  Ack: 0xFB8BDE13  Win: 0x832C  TcpLen: 20
3C 48 54 4D 4C 3E 0A 3C 48 45 41 44 3E 0A 3C 53  <HTML>.<HEAD>.<S
43 52 49 50 54 20 4C 41 4E 47 55 41 47 45 3D 22  CRIPT LANGUAGE="
4A 61 76 61 53 63 72 69 70 74 31 2E 31 22 3E 0A  JavaScript1.1">.
3C 21 2D 2D 0A 64 6F 63 75 6D 65 6E 74 2E 6C 6F  <!--.document.lo
63 61 74 69 6F 6E 2E 72 65 70 6C 61 63 65 28 22  cation.replace("
68 74 74 70 3A 2F 2F 68 6F 6D 65 2E 6E 65 74 73  http://home.nets
63 61 70 65 2E 63 6F 6D 2F 65 78 2F 73 68 61 6B  cape.com/ex/shak
2F 69 6E 64 65 78 2E 68 74 6D 6C 22 29 3B 0A 2F  /index.html");./
2F 2D 2D 3E 0A 3C 2F 53 43 52 49 50 54 3E 0A 3C  /-->.</SCRIPT>.<
4D 45 54 41 20 48 54 54 50 2D 45 51 55 49 56 3D  META HTTP-EQUIV=
22 52 65 66 72 65 73 68 22 20 43 4F 4E 54 45 4E  "Refresh" CONTEN
54 3D 22 30 3B 20 55 52 4C 3D 68 74 74 70 3A 2F  T="0; URL=http:/
2F 68 6F 6D 65 2E 6E 65 74 73 63 61 70 65 2E 63  /home.netscape.c
6F 6D 2F 65 78 2F 73 68 61 6B 2F 69 6E 64 65 78  om/ex/shak/index
2E 68 74 6D 6C 22 3E 0A 3C 2F 48 45 41 44 3E 0A  .html">.</HEAD>.
3C 2F 48 54 4D 4C 3E 0A 0A                       </HTML>..
```

7. Evidence of active targeting:
Yes, (if my hair brained ideas are right) this type of activity when on for about a week solid before I applied ACLs

8. Severity:

Target Criticality: **2**, This attack only seems to be focused on client systems that browse the web.

Attack Lethality: **4**, This attack could be a DDoS or an attemp to get passwords though Cookies.

System Countermeasures: **5,** All machines are up to date with security patches

Network Countermeasures: **2,** Firewall is up to date and the only way in or out, however none of this traffic seemed to get logged

(2 + 4) – (5 + 2) = -1

9. Defensive recommendation:
Since none of the inbound traffic was recorded by our firewall I set ACLs on our router for both inbound and outbound traffic.

10. Multiple choice test question:

What is the common IDS decode filter for the Unicode exploit

a. **| 07|**
b. ******
c. **../** =correct
d. **%sys%**

Assignment 2 - Describe the State of Intrusion Detection (30 Points)

Sub Seven 2.2

In an ever-changing world, the use of computers in everyday life just seems to rise without any sign of backing down. In our homes, our office, our cars and even in our wallets and pocket books, we have any number of computer related products which add such enhancement to our lives that we are at point where life without computers would be less convenient than we would like to think. One of the best sources that add to this enhancement or "convenience is the internet. It allows us to communicate with family and friends, find otherwise impossible to find information, and makes it possible to carry out business transactions in a quick and easy manner, in short the internet has made connecting to the entire world at any time a reality. Along with all of the good things the Internet can provide for us, there is a flip side as well; Hackers! Just as there are many tools to help us, for the majority: the computer novice, there are also many tools to help the hacker. For the most part these tools are pre-made, out-of-the-box solutions just like our word processors, e-mail and graphics programs. Anyone who wishes to down load these hacker tools can be a "real" hacker in a matter of hours, if fact the majority of all hackers are just that, what we call "script-kiddies". One of the most popular any most methods that these script kiddies use to install a backdoor or opening on your network is

the “Trojan Horse”. Usually a Trojan is sent through e-mail as an attachment which is very easily executed by the by the unknowing user if for nothing but pure interest such as a file named “prettylady.exe”. These files once executed can perform a wide number of functions like deleted files, opening ports for access to the hacker of just creating a general nuisance. These files can also be spread to other systems through e-mail. SubSeven, created around May of 1999 has quickly gain ground as one of the most common and powerful tools of choice.

In a Nutshell

SubSeven is a remote access and control program that can perform a wide variety of functions ranging from common annoyances to full blown compromise of files and critical data on the infected host(s). Since its creation by a hacker who calls himself Mobman, SubSeven has gone through many version changes and now is at its latest release: Version2.2. There are three main components to this new version; the server, the edit server program, and the client. In order for this all to work, first the hacker must install the server on the host(s) of choice, this is commonly done through e-mail file attachments. This server file can be customized through the use of the Edit Server program, in which the hacker can change many parameters such as server port number, installation methods, executable name, methods of notifying the hacker that this host is online and many others (described in detail in “The Components” section). Once the server portion is successfully installed on the host the hacker can now use the SubSeven client to attach to the host via the port that was pre determined by the Edit Sever program (27374 by default). Now the fun begins, the hacker can perform many tasks on the victim host ranging from changing hardware settings, (changing window colors, opening and closing the cd-rom, reversing the mouse buttons, rebooting the computer, etc.), to information gathering, (windows version, user’s name and address, hard drive and file information, and password information) and installing program updates to the infected host. The SubSeven sever host is not the only victim here the hackers can also scan other networks and perform DDoS attacks from the infected host, all the while keeping his identity and involvement a secret. New features in version 2.2 include support for socks proxies, a packet sniffer, random port listening (notifications of changes are sent to hacker), CGI notifications and the ability to send keystrokes to remote system(s).

The Components

SubSeven Server

This is the program that must be installed on the victim’s computer for this tool to work, it is typically found in two places on windows 9x PC’s, in the \Windows directory as srv32.exe (by default, but can be customized as any name by the edit server program), in the \Windows\system directory as RunDLL32.dll (by default, but can be customized as any name by the edit server program). Also, the sever can write registry, win.ini and system.ini settings to the victim computer in the following places:

WIN.INI "load=" or "run=" line

SYSTEM.INI. "shell=" line

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices

EditServer

This is a GUI program that allows the hacker to customize the server portion prior to infecting the target host. The EditServer program can manipulate almost any setting possible for the server program; the server settings section allows the hacker to change the default port, passwords and the file name for the server program. In the startup methods section, the dll name can be set as well as the way in which the server program is installed, next we have the notifications section where one can choose the way in which the attacker is notified when the host is online and ready to be accessed. The binded files and plugings section allow the hacker to include SubSeven pulgings as well as external programs to run on the infected host. Additional settings include; an e-mail section where logged keys and password can be e-mailed to the hacker, a restrictions section that can limit the commands allowed on the server and a section that allows the creation of custom errors and the look of the server icon.

Subseven Client

The client is where all the fun begins; this is the main GUI console where the hacker can perform such a large number remote commands it is as if the hacker is sitting at the victims computer locally. The commands are grouped together on the left hand side of the program in "explorer" style fashion with a command window on the right. The commands are grouped into the following:

SHORTCUTS- This command group is as it sounds, simple shortcuts to popular commands found in other command groups.

CONNECTION- Within this group is the bulk of the information gathering tools such as pc info (2) and home info, which can be used to gather just about anything about the victim from pc info to personal info about the user. This group also contains the built-in scanner that can be used to scan remote or local host while hiding the true source of the attacker. Server options are included to allow the remote manipulation of the server settings on the victim such as port, password and updates. Other commands in this group include proxy support, connection commands and web status/download.

KEYS/MESSAGES- Complete control over the keyboard of the victim, pc is the main part of this group, which included a key logger, key sender and manipulation of keys. ICQ takeover, which can take the identity of the ICQ user, also resides in this group.

ADVANCED- Ranging from FTP services to registry editor, this group is yet another major, and most dangerous, collection of commands for controlling almost anything on the compromised host. The find files and the network browser commands operates just like its windows counterparts that can search for any file, which resides on the host or network. Password gathering utilities for almost any application can be found here, as well as application and port redirectors. A built-in packet sniffer can be used to capture traffic on the host network or the Internet.

MISCELLANEOUS-Various commands are included here, but perhaps the best is a text-2-speech application that converts typed words to speech that is outputted to the host's pc speakers.

FUN MANAGER & FUN OTHER- These commands fall under the "nuisance" category with such options as reverse mouse buttons, flip screen, reset multi-media commands and even a tic-tac-toe game. Screen captures and a web cam recorded are included as well in this group.

PLUGINS- This is the area where installed plug-ins are managed.

LOCAL OPTIONS- IP tools, an address book of compromised systems and preferences are some of the commands that reside here.

Network trace examples

Searching for SubSeven open ports (from my snort system)

```
Feb 16 22:03:47 141.150.211.31:1920 -> x.x.x.2:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1921 - > x.x.x.3:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1922 -> x.x.x.4:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1923 -> x.x.x.5:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1925 -> x.x.x.7:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1929 -> x.x.x.11:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1930 -> x.x.x.12:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1933 -> x.x.x.15:27374 SYN ******S*
Feb 16 22:03:47 141.150.211.31:1936 -> x.x.x.18:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:1948 -> x.x.x.30:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:1979 -> x.x.x.61:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:1984 -> x.x.x.66:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:1987 -> x.x.x.69:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:1998 -> x.x.x.80:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:2017 -> x.x.x.99:27374 SYN ******S*
Feb 16 22:03:48 141.150.211.31:2018 -> x.x.x.100:27374 SYN ******S*
Feb 16 22:03:49 141.150.211.31:2045 -> x.x.x.127:27374 SYN ******S*
```

Log file example from Tlsecurity:

```
596     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\SRV32.EXE      SUCCESS          Offset: 37888
Length: 4096
597     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\SRV32.EXE      SUCCESS          Offset: 41984
Length: 4096
598     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\SRV32.EXE      SUCCESS          Offset: 46080
Length: 4096
599     18:36:58  Srv32    Attributes  C:\WINDOWS\SYSTEM\SRV32.DEL      NOTFOUND         GetAttributes
600     18:36:58  Srv32    Attributes  C:\WINDOWS\SYSTEM\SRV32.DEL.DLL          NOTFOUND
        GetAttributes
601     18:36:58  Srv32    Attributes  C:\WINDOWS\SYSTEM\SRV32.DE       NOTFOUND         GetAttributes
602     18:36:58  Srv32    Attributes  C:\WINDOWS\SYSTEM\SRV32.DE.DLL   NOTFOUND         GetAttributes
603     18:36:58  Srv32    Attributes  C:\PROGRAM FILES\AMI SCROLL\4DMAIN.EXE SUCCESS
        GetAttributes
604     18:36:58  Srv32    Directory   C:\PROGRAM FILES\AMI SCROLL\4DMAIN.EXE SUCCESS          QUERY
605     18:36:58  Srv32    Read        C:\PROGRA~1\AMISCR~1\4DMAIN.EXE          SUCCESS          Offset:
30720 Length: 4096
606     18:36:58  Srv32    Attributes  C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS
        GetAttributes
607     18:36:58  Srv32    Directory   C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS          QUERY
608     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS          Offset:
39936 Length: 2560
609     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS          Offset:
39936 Length: 2560
610     18:36:58  Srv32    Read        C:\PROGRA~1\AMISCR~1\4DMAIN.EXE          SUCCESS          Offset:
30720 Length: 4096
611     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS          Offset:
31232 Length: 4096
612     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS          Offset:
39424 Length: 512
613     18:36:58  Srv32    Read        C:\WINDOWS\SYSTEM\4DHOOK32.DLL           SUCCESS          Offset:
35328 Length: 4096
614     18:36:58  Srv32    Attributes  C:\WINDOWS\SYSTEM\RNR20.DLL      SUCCESS          GetAttributes
```

Playing Clean-up

Removing SubSeven2.2 can be both easy and difficult at the same time; the easy part is standard location the win.ini, system.ini and registry entries but, in the nature of this exploit the files placed on the host machine can be named virtually anything making it difficult to find these files. Making it a little easier, for the most part the “server.exe” is usually 328kb in size. In an effort to outline the basics of removal of SubSeven, the following will note the “out of the box” names of the sever program.

First remove files found in these locations:

\WINDOWS\ srv32.exe

\WINDOWS\SYSTEM\ rundll32.dll

Next examine the these ini files for the following entries and remove them

WIN.INI - load= srv32.exe or run= srv32.exe

SYSTEM.INI. - shell=Explorer.exe srv32.exe

Lastly, remove the follwing registry entries:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices

Assignment 3 - "Analyze This" Scenario (30 Points)

For this portion of the SANS practical, we have been asked to analyze over a months worth of Snort data collected from a client's network. Using a standard snort rule base the data was split into three types; Alerts, Scans and log files. We know that due to uncontrollable circumstances, (power failure, full disk drives, etc.) some of the data is not complete.

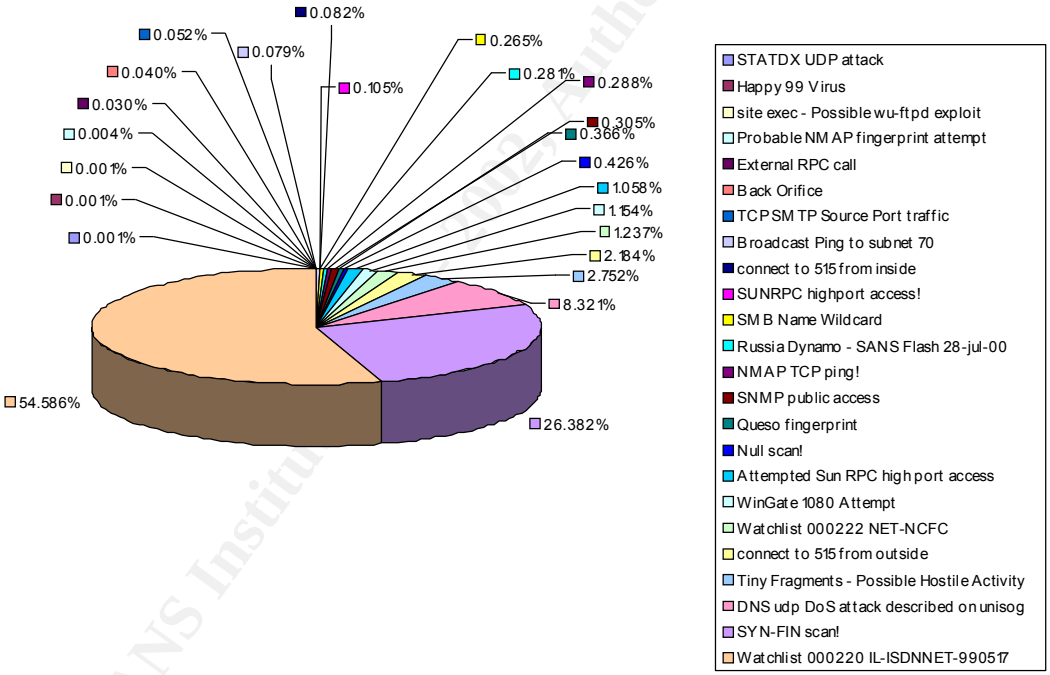
ALERT FILES

First we will look at the 24 unique alert data in the files given to us in ascending frequency.

STATDX UDP attack	1
Happy 99 Virus	1
site exec - Possible wu-ftpd exploit	2
Probable NMAP fingerprint attempt	8
External RPC call	59
Back Orifice	77
TCP SMTP Source Port traffic	100
Broadcast Ping to subnet 70	154
connect to 515 from inside	159
SUNRPC highport access!	204
SMB Name Wildcard	515
Russia Dynamo - SANS Flash 28-jul-00	546
NMAP TCP ping!	558
SNMP public access	591
Queso fingerprint	710

Null scan!	826
Attempted Sun RPC high port access	2,053
WinGate 1080 Attempt	2,239
Watchlist 000222 NET-NCFC	2,401
connect to 515 from outside	4,238
Tiny Fragments - Possible Hostile Activity	5,340
DNS udp DoS attack described on unisog	16,146
SYN-FIN scan!	51,192
Watchlist 000220 IL-ISDNNET-990517	105,918

Next, we divide up those unique alerts on a percentage basis



Lets take a closer look at the most hostile and intrusive attacks:

Watchlist 000220 IL-ISDNNET-990517

The main concern with this alert is the sheer percentage of alerts; many different ports and sources of concern are present here

To port 23

01/07-03:52:17.757818 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.12:49089 -> MY.NET.60.11:23

01/07-03:52:17.937671 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.12:49089 -> MY.NET.60.11:23

01/07-03:52:18.315168 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.12:49089 -> MY.NET.60.11:23

01/07-03:52:18.494859 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.12:49089 -> MY.NET.60.11:23

01/07-03:52:18.544149 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.12:49089 -> MY.NET.60.11:23

01/07-03:52:18.674344 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.12:49089 -> MY.NET.60.11:23

From port 1

12/28-19:18:35.257891 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.44.105:1 -> MY.NET.130.187:2209

12/28-19:18:35.586618 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.44.105:1 -> MY.NET.130.187:2209

12/28-19:18:35.753331 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.44.105:1 -> MY.NET.130.187:2209

12/28-19:18:36.134957 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.44.105:1 -> MY.NET.130.187:2209

12/28-19:18:37.635936 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.44.105:1 -> MY.NET.130.187:2209

12/28-19:18:38.220119 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.44.105:1 -> MY.NET.130.187:2209

To port 443

01/07-00:03:58.221209 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.42.102:1164 -> MY.NET.5.29:443

01/07-00:04:03.582632 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.42.102:1165 -> MY.NET.5.29:443

01/07-00:04:04.095732 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.42.102:1165 -> MY.NET.5.29:443

01/07-00:04:04.121337 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.42.102:1165 -> MY.NET.5.29:443

01/07-00:04:06.257057 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.42.102:1168 -> MY.NET.5.29:443

01/07-00:04:06.324086 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.42.102:1166 -> MY.NET.5.29:443

To port 25

01/04-09:43:01.272515 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.4:38260 -> MY.NET.253.42:25

01/04-09:44:16.383994 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.4:38284 -> MY.NET.253.43:25

01/04-09:44:16.572307 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.4:38284 -> MY.NET.253.43:25

01/04-09:44:26.899888 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.4:38284 -> MY.NET.253.43:25

01/04-09:44:27.092505 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.4:38284 -> MY.NET.253.43:25

01/04-09:44:27.465986 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.58.4:38284 -> MY.NET.253.43:25

Tiny Fragments - Possible Hostile Activity

Fragmentation is commonly used by hackers to confuse, over load and crash systems, the most famous of these attacks is Teardrop.

01/03-04:14:04.035727 [**] Tiny Fragments - Possible Hostile Activity [**] 202.108.43.152 -> MY.NET.1.8

01/03-04:31:26.420489 [**] Tiny Fragments - Possible Hostile Activity [**] 202.108.43.152 -> MY.NET.1.8

01/03-04:31:26.420543 [**] Tiny Fragments - Possible Hostile Activity [**] 202.108.43.152 -> MY.NET.1.8

01/03-04:50:36.515983 [**] Tiny Fragments - Possible Hostile Activity [**] 202.108.43.152 -> MY.NET.1.8

01/03-04:50:36.516105 [**] Tiny Fragments - Possible Hostile Activity [**] 202.108.43.152 -> MY.NET.1.8

01/03-05:03:58.703988 [**] Tiny Fragments - Possible Hostile Activity [**] 202.108.43.152 -> MY.NET.1.8

WinGate 1080 Attempt

WinGate 1080 is typically an attempt to probe or compromise proxy servers or socks ports.

12/16-15:01:51.421734 [**] WinGate 1080 Attempt [**] 194.72.6.103:51954 -> MY.NET.60.38:1080

12/16-15:28:39.078857 [**] WinGate 1080 Attempt [**] 207.114.4.46:1528 -> MY.NET.98.168:1080

12/16-15:39:07.529352 [**] WinGate 1080 Attempt [**] 204.117.70.5:2741 -> MY.NET.204.174:1080

12/16-15:39:07.579137 [**] WinGate 1080 Attempt [**] 204.117.70.5:2743 -> MY.NET.204.174:1080

12/16-15:42:51.485357 [**] WinGate 1080 Attempt [**] 204.117.70.5:2960 -> MY.NET.211.142:1080

12/16-15:42:52.988021 [**] WinGate 1080 Attempt [**] 204.117.70.5:2964 -> MY.NET.211.142:1080

12/16-17:52:00.300992 [**] WinGate 1080 Attempt [**] 194.84.208.118:3502 -> MY.NET.60.38:1080

Back Orifice

Back orifice is perhaps the most commonly known “back door” Trojan and is very dangerous in the fact that a hacker can take almost total control of a compromised host.

Evidance of infected host:

12/09-22:23:21.448457 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.47:31337

12/09-22:23:21.525455 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.56:31337

12/09-22:23:21.707252 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.154:31337

12/09-22:23:21.707459 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.156:31337

12/09-22:23:21.710371 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.155:31337

12/09-22:23:21.898431 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.172:31337

12/09-22:23:21.911339 [**] Back Orifice [**] 209.94.199.202:31338 -> MY.NET.60.174:31337

site exec - Possible wu-ftpd exploit

This alert is particularly dangerous because it may represent possible root comprimise

12/21-15:26:29.595664 [**] site exec - Possible wu-ftpd exploit - GIAC000623 [**] 64.217.116.106:1684 -> MY.NET.97.162:21

11/26-17:30:50.939661 [**] site exec - Possible wu-ftpd exploit - GIAC000623 [**] 24.23.255.246:4507 -> MY.NET.130.98:21

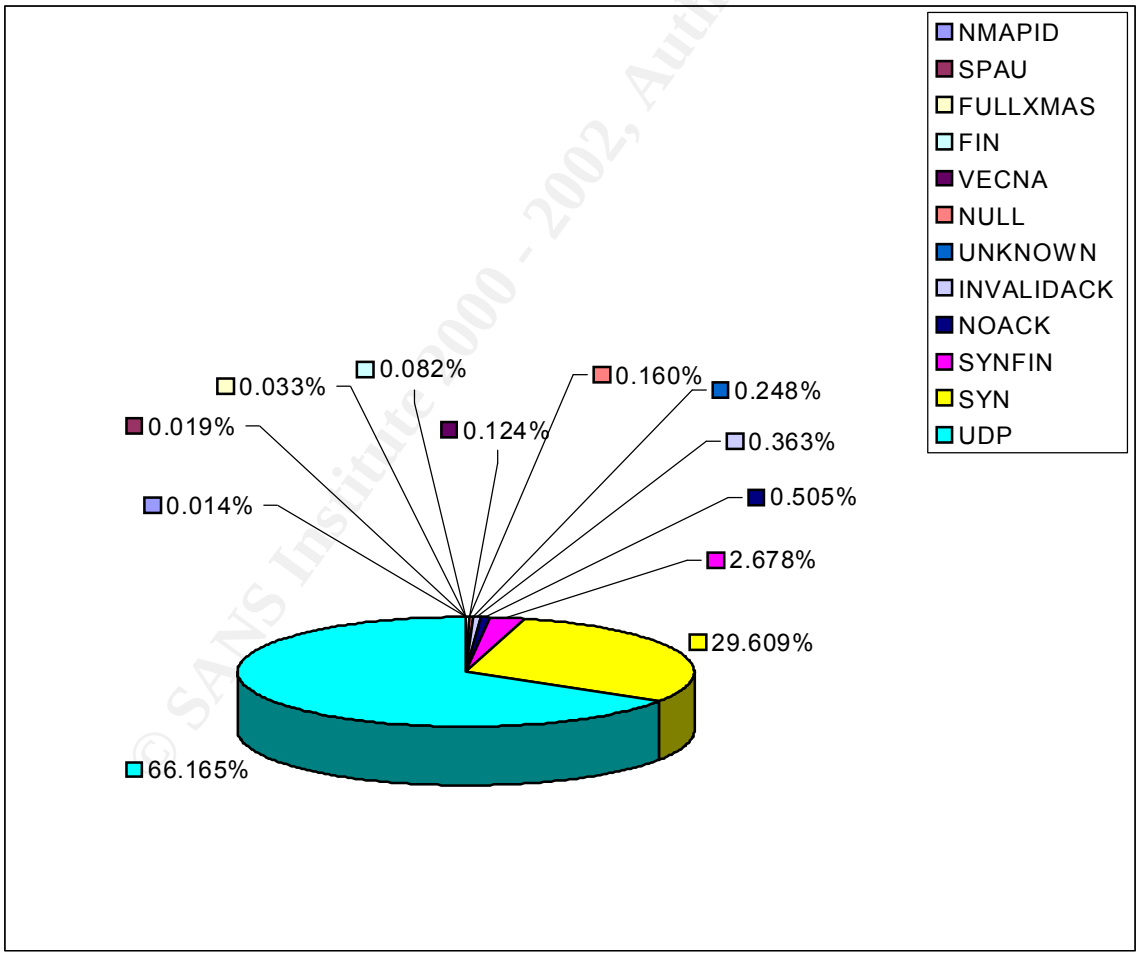
SCAN FILES

These reports represent scans that where collected on the clients network

NMAPID	122
SPAU	163
FULLXMAS	280
FIN	698
VECNA	1,060
NULL	1,368

UNKNOWN	2,121
INVALIDACK	3,102
NOACK	4,314
SYNFIN	22,902
SYN	253,185
UDP	565,769

And the scan in percentage format:



Most interesting scans:

UDP port 0 to 0

Dec 21 00:28:39 MY.NET.97.206:0 -> 172.142.147.21:0 UDP

Dec 21 00:28:44 MY.NET.97.206:0 -> 24.66.133.31:0 UDP

Dec 21 00:28:46 MY.NET.97.206:0 -> 210.94.64.44:0 UDP

Possible Back construction compromise:

Dec 21 22:09:39 MY.NET.214.166:666 -> 24.24.42.137:62298 UDP

Dec 21 22:09:40 MY.NET.214.166:666 -> 64.128.132.105:27961 UDP

Dec 21 22:09:40 MY.NET.214.166:666 -> 209.90.4.89:26650 UDP

Possible God message compromise:

Dec 8 14:28:46 MY.NET.71.38:7777 -> 200.38.23.236:1033 UDP

Dec 8 14:28:46 MY.NET.71.38:7777 -> 129.2.223.27:1029 UDP

Dec 8 14:28:46 MY.NET.71.38:7777 -> 63.193.188.111:64706 UDP

Possible Netministrator compromise:

Jan 1 06:40:46 MY.NET.202.94:9000 -> 207.46.204.95:9004 UDP

Jan 1 06:40:46 MY.NET.202.94:9000 -> 207.46.204.86:9000 UDP

Jan 1 06:40:47 MY.NET.202.94:9000 -> 207.46.204.98:9000 UDP

Several other scans that use fingerprinting attempts and network mapping such as SYN, SYN/FIN, FIN and the use of reserved bit scans like SPAU and FULLXMAS where present on the network during the time.

Analysis methods

This portion of the practical was VERY difficult for me for the simple fact that I ran all my tests on a Windows 2000 laptop, which was already heavily loaded. With the combination of this and my lack of knowledge of writing custom perl scripts to sort data, I was forced to go with my own resources, doing it pretty much all back hand and eye!

First, I compiled the individual Alert and scan files into one text file of each respectively

Second, I imported these files into MS Access and sorted by the Alert message or Scan type and cut and pasted these by type back into separate spreadsheets to get total and individual entry numbers.

Third, I created MS Excel spreadsheets with type and number of attacks/scans and used this to create percentage graphs.

The rest was just looking through the broken up data and pulling out what I saw as interesting traffic.

References:

Sam spade version1.14 by Steve Atkins
http://www.samspade.org/ssw/

Deconstructing SubSeven, the Trojan Horse of Choice
Jamie Crapanzano, January 8, 2001
http://www.sans.org/infosecFAQ/malicious/subseven.htm

What is SubSeven? Giving away control of your machine!
James Wentzel February 16, 2001
http://www.sans.org/infosecFAQ/malicious/subseven2.htm

The Basics of SubSeven (aka Sub7 or Backdoor_G) no author
http://www.commodon.com/threat/threat-sub7.htm

SubSeven version 2.2 obtiained at
http://www.tlsecurity.net/main.htm

Internet Security Systems Security Alert October 8, 2000
http://xforce.iss.net/alerts/advise65.php

Internet Security Systems Security Alert March 12, 2001
http://xforce.iss.net/alerts/advise73.php

Trojan list at:
http://www.simovits.com/nyheter9902.html