



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Intrusion Detection In-Depth (Security 503)"
at <http://www.giac.org/registration/gcia>

GCIA 2.9

**Intrusion Detection In Depth
SANS Parliament Square
London 2001**

Simon Devlin

© SANS Institute 2000 - 2002, Author retains full rights.

Contents

1	Assignment #1 - Detects	4
1.1	Detect #1 - "Code Red Worm" to Linux Apache Server	5
1.1.1	Trace Data	5
1.1.2	Location Of Trace	6
1.1.3	Source Of Trace	6
1.1.4	Likelihood Of Address Spoofing?	7
1.1.5	Description Of Attack	7
1.1.6	Attack Mechanism	7
1.1.7	Correlations	7
1.1.8	Evidence Of Active Targeting	8
1.1.9	Severity	8
1.1.10	Defensive Recommendations	8
1.1.11	Test Question	8
1.2	Detect #2 - Web services portscan against address space.	9
1.2.1	Trace Data	9
1.2.2	Location Of Trace	12
1.2.3	Source Of Trace	12
1.2.4	Likelihood Of Address Spoofing?	12
1.2.5	Description Of Attack – NOT COMPLETED	12
1.2.6	Attack Mechanism	12
1.2.7	Correlations	13
1.2.8	Evidence Of Active Targeting	13
1.2.9	Severity	14
1.2.10	Defensive Recommendations	14
1.2.11	Test Question	14
1.3	Detect #3 - WU-FTPD "SITE EXEC" exploit	15
1.3.1	Trace Data	15
1.3.2	Location Of Trace	15
1.3.3	Likelihood Of Address Spoofing?	15
1.3.4	Description Of Attack	16
1.3.5	Attack Mechanism	16
1.3.6	Correlations	16
1.3.7	Evidence Of Active Targeting	16
1.3.8	Severity	16
1.3.9	Defensive Recommendations	17
1.3.10	Test Question	17
1.4	Detect #4 – Bulk CGI vulnerability scan	17
1.4.1	Trace Data	17
1.4.2	Location Of Trace	18
1.4.3	Likelihood Of Address Spoofing?	18
1.4.4	Description Of Attack	18
1.4.5	Attack Mechanism	18
1.4.6	Correlations	18
1.4.7	Evidence Of Active Targeting	19
1.4.8	Severity	19
1.4.9	Defensive Recommendations	19
1.4.10	Test Question	20
1.5	Detect #5 – Low Port-Low Port SYNFIN Scan	20

1.5.1	Trace Data	20
1.5.2	Location Of Trace	21
1.5.3	Likelihood Of Address Spoofing?	21
1.5.4	Description Of Attack	21
1.5.5	Attack Mechanism	21
1.5.6	Correlations.....	21
1.5.7	Evidence Of Active Targeting	22
1.5.8	Severity	22
1.5.9	Defensive Recommendations	22
1.5.10	Test Question	22
2	Assignment #2 - What happens before and after your sensor alarms?	23
2.1	Introduction	23
2.2	Method.....	23
2.3	What Do you Need?	25
2.3.1	The Hub.....	25
2.3.2	The Archiver	25
2.3.3	Your Sensor	26
2.4	Task List.....	26
2.5	Tuning.....	26
2.5.1	Packet Sizes	26
2.5.2	Encrypted Traffic	27
2.5.3	Bulk Traffic.....	27
2.6	Relevant Scripts	27
2.6.1	Script To Start At Boot	27
2.6.2	Crontab entry to Rotate TCPDUMP files	28
2.6.3	Script to Rotate TCPDUMP files	28
2.7	Conclusion.....	28
2.8	References	29
3	Assignment #3 - Analyse This	30
3.1	List of Analysed Files	30
3.2	Overview	30
3.2.1	Multicast.....	30
3.2.2	Trojan Activity	30
3.2.3	Hostile Sources	30
3.3	Prioritised Detects	31
3.3.1	Possible trojan server activity	31
3.3.2	Watchlist 000220 IL -ISDNNET-990517.....	32
3.3.3	External RPC call	32
3.3.4	SMB Name Wildcard	32
3.3.5	connect to 515 from outside	32
3.3.6	UDP SRC and DST outside network	32
3.3.7	Watchlist 000222 NET -NCFC	32
3.3.8	Queso fingerprint.....	33
3.3.9	Back Orifice	33
3.3.10	High port 65535 tcp - possible Red Worm - traffic	33
3.4	Top Talkers	33
3.4.1	Port Scans.....	33
3.4.2	Alerts	34
3.4.3	Out Of Spec	35
3.5	External Source Registration Details	36

3.5.1	212.179.79.2	36
3.5.2	129.170.104.19	36
3.5.3	216.220.164.141	37
3.5.4	165.230.53.35	37
3.5.5	211.152.241.1	38
3.5.6	24.147.14.159	38
3.5.7	159.226.41.166	39
3.5.8	216.220.167.94	39
3.5.9	24.27.62.134	40
3.6	Correlation's	40
3.6.1	Possible trojan server activity	40
3.6.2	Watchlist 000220 IL -ISDNNET-990517.....	40
3.6.3	SMB Name Wildcard	41
3.6.4	connect to 515 from outside	41
3.6.5	UDP SRC and DST outside network	41
3.6.6	Watchlist 000222 NET -NCFC	41
3.6.7	Queso fingerprint.....	41
3.6.8	Back Orifice	41
3.6.9	High port 65535 tcp - possible Red Worm - traffic	42
3.7	Out Of Spec (OOS) Link Analysis	42
3.8	Anomalous Activity	42
3.8.1	Sub7.....	42
3.8.2	High port 65535 tcp - possible Red Worm	43
3.8.3	Source / Destination Port 0	43
3.8.4	Probable ECN TCP Flags	43
3.8.5	Unknown TCP Options	43
3.8.6	SYN FIN Flags	44
3.8.7	XMAS Packets	44
3.9	Defensive Recommendations	44
3.9.1	Ingress / Egress Filtering	44
3.9.2	Anti-Spoofing	45
3.9.3	Restrictive Firewall Policy	45
3.9.4	Email Attachment Scanning	46
3.9.5	Virus Scanning	46
3.10	Analysis Process	46
4	Appendices	48
4.1	Apache Failed Document Script	48
4.2	Non-Multicast Alert Breakdowns	48
4.2.1	Alert Summary	48
4.2.2	Top Sources	49
4.2.3	Top Destinations.....	49

Figures

Figure 1	- incidents.org August 1st Code Red progress	8
Figure 2	- Firewall log showing dropped web service connections	13
Figure 3	- Snortsnarf Summary	19
Figure 4	- Sub7 configuration	31
Figure 5	- OOS Link Analysis for top source	42

1 Assignment #1 - Detects

1.1 Detect #1 - "Code Red Worm" to Linux Apache Server

1.1.1 Trace Data

1.1.1.1 Snort Alert Log

```
[**] WEB-IIS ISAPI .ida attempt [**]  
08/01-06:37:23.420000 61.113.0.16:10027 -> badger:80  
TCP TTL:240 TOS:0x10 ID:0 IpLen:20 DgmLen:1504  
***AP*** Seq: 0x4A3730BD Ack: 0xCA348D2E Win: 0x7FE0 TcpLen: 20  
0x0000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0160: 27 2B 00 50 4A 37 30 BD CA 34 8D 2E 50 18 7F E0 '+.PJ70..4..P..  
0x0170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x01A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x01B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x01C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x01D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x01E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x01F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x0290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x02A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x02B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x02C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x02D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x02E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
0x02F0: 47 45 54 20 2F 64 65 66 61 75 6C 74 2E 69 64 61 GET /default.ida  
0x0300: 3F 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E ?NNNNNNNNNNNNNNNN  
0x0310: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0320: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0330: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0340: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0350: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0360: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0370: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0380: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x0390: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x03A0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN  
0x03B0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E NNNNNNNNNNNNNNNNN
```

```

0x03C0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNNN
0x03D0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNNN
0x03E0: 4E 25 75 39 30 39 30 25 75 36 38 35 38 25 75 63  N%u9090%u6858%uc
0x03F0: 62 64 33 25 75 37 38 30 31 25 75 39 30 39 30 25  bd3%u7801%u9090%
0x0400: 75 36 38 35 38 25 75 63 62 64 33 25 75 37 38 30  u6858%ucbd3%u780
0x0410: 31 25 75 39 30 39 30 25 75 36 38 35 38 25 75 63  1%u9090%u6858%uc
0x0420: 62 64 33 25 75 37 38 30 31 25 75 39 30 39 30 25  bd3%u7801%u9090%
0x0430: 75 39 30 39 30 25 75 38 31 39 30 25 75 30 30 63  u9090%u8190%u00c
0x0440: 33 25 75 30 30 30 33 25 75 38 62 30 30 25 75 35  3%u0003%u8b00%u5
0x0450: 33 31 62 25 75 35 33 66 66 25 75 30 30 37 38 25  31b%u53ff%u0078%
0x0460: 75 30 30 30 30 25 75 30 30 3D 61 20 20 48 54 54  u0000%u00=a HTT
0x0470: 50 2F 31 2E 30 0D 0A 43 6F 6E 74 65 6E 74 2D 74  P/1.0..Content-t
0x0480: 79 70 65 3A 20 74 65 78 74 2F 78 6D 6C 0A 48 4F  ype: text/xml.HO
0x0490: 53 54 3A 77 77 77 2E 77 6F 72 6D 2E 63 6F 6D 0A  ST:www.worm.com.
0x04A0: 20 41 63 63 65 70 74 3A 20 2A 2F 2A 0A 43 6F 6E  Accept: */*.Con
0x04B0: 74 65 6E 74 2D 6C 65 6E 67 74 68 3A 20 33 35 36  tent-length: 356
0x04C0: 39 20 0D 0A 0D 0A 55 8B EC 81 EC 18 02 00 00 53  9 ....U.....S
0x04D0: 56 57 8D BD E8 FD FF FF B9 86 00 00 00 B8 CC CC  VW.....
0x04E0: CC CC F3 AB C7 85 70 FE FF FF 00 00 00 E9 0A  ....p.....
0x04F0: 0B 00 00 8F 85 68 FE FF FF 8D BD F0 FE FF FF 64  ....h.....d
0x0500: A1 00 00 00 00 89 47 08 64 89 3D 00 00 00 00 E9  ....G.d.=.....
0x0510: 6F 0A 00 00 8F 85 60 FE FF FF C7 85 F0 FE FF FF  o.....
0x0520: FF FF FF FF 8B 85 68 FE FF FF 83 E8 07 89 85 F4  ....h.....
0x0530: FE FF FF C7 85 58 FE FF FF 00 00 E0 77 E8 9B 0A  ....X.....w...
0x0540: 00 00 83 BD 70 FE FF FF 00 0F 85 DD 01 00 00 8B  ....p.....
0x0550: 8D 58 FE FF FF 81 C1 00 00 01 00 89 8D 58 FE FF  .X.....X..
0x0560: FF 81 BD 58 FE FF FF 00 00 00 78 75 0A C7 85 58  ...X.....xu...X
0x0570: FE FF FF 00 00 F0 BF 8B 95 58 FE FF FF 33 C0 66  ....X...3.f
0x0580: 8B 02 3D 4D 5A 00 00 0F 85 9A 01 00 00 8B 8D 58  .=MZ.....X
0x0590: FE FF FF 8B 51 3C 8B 85 58 FE FF FF 33 C9 66 8B  ....Q<..X...3.f.
0x05A0: 0C 10 81 F9 50 45 00 00 0F 85 79 01 00 00 8B 95  ....PE....y....
0x05B0: 58 FE FF FF 8B 42 3C 8B 8D 58 FE FF FF 8B 54 01  X...B<..X...T.
0x05C0: 78 03 95 58 FE FF FF 89 95 54 FE FF FF 8B 85 54  x..X....T....T
0x05D0: FE FF FF 8B 48 0C 03 8D 58 FE FF FF 89 8D 4C FE  ....H...X....L.
0x05E0: FF FF 8B 95 4C FE FF FF 81 3A 4B 45 52 4E  ....L....:KERN

```

====+

1.1.2 Location Of Trace

Home internet connection – an ADSL connection providing a fixed /28 address range.

1.1.3 Source Of Trace

Snort 1.8 (build 43). I have a low end Cisco Catalyst switch and use the SPAN port functionality to copy frames from the port attached to the ADSL router to a second 10BaseT switch port. This is attached to a dedicated interface monitored by Snort. To minimize risk, this interface has no IP configuration associated with it (as shown below).

```

[devlinse@grunt /]$ ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:01:02:F3:7B:B4
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1019468 errors:0 dropped:0 overruns:1 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:3 Base address:0xc800

[devlinse@grunt /]$

```

To make it clear, the frames are copied to Snort prior to them hitting the Firewall.

Given that the internet connection is DSL with an upper limit of 512kbps and that the machines performing IDS duties are dedicated to this task, the chances of missing packets through network loading are low.

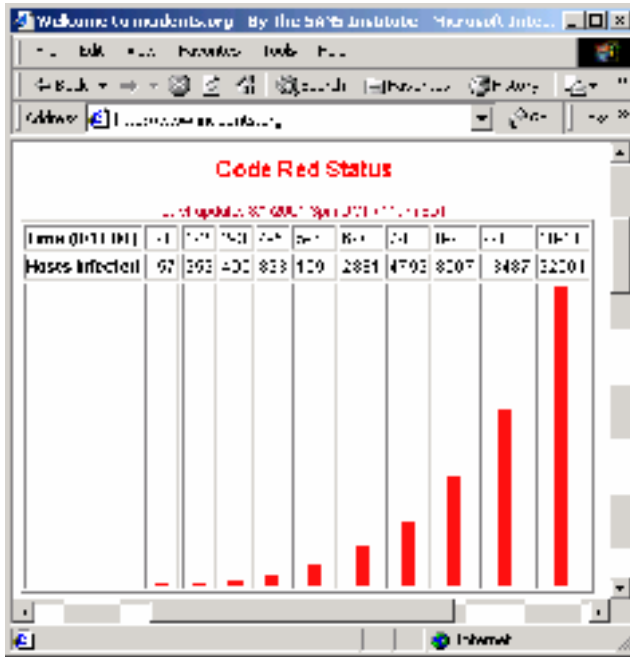


Figure 1 - incidents.org August 1st Code Red progress

1.1.8 Evidence Of Active Targeting

None. The webserver (badger) that was the target is a Unix based Apache server and not susceptible to the worm.

1.1.9 Severity

Criticality	4	Target is a webserver
+ Lethality	1	Target is wrong platform / operating system for attack to succeed.
-		
System Countermeasures	3	Old Redhat box
+ Network Countermeasures	1	No firewall.
Total	1	Low

1.1.10 Defensive Recommendations

The severity value above is adversely affected by the results for the System and Network countermeasure components. For reference, this server was specifically built with Redhat 5.2, a number of exposed services (such as RPC) and placed outside of a firewall to act as a kind of ho ney-pot. The fact that it picked up “Code Red” was just a side effect.

Given that this particular attack cannot succeed against an Apache webserver, there are no real defensive recommendations to be made regarding this example.

1.1.11 Test Question

Given the contents of the Snort log, what can we say about this request?

- It is likely to be CodeRed V1
- It is likely to be CodeRed V2

- c) The host being attacked is www.host.com
- d) This is a variation using HTTP1.1 as a transport.

Answer

- a. A major part of the payload changed from N's to X's on the 2nd release.

1.2 Detect #2 - Web services portscan against address space.

1.2.1 Trace Data

The data below is repeated for every address in the /28 range, but is omitted for brevity. Please note that destination address have been replaced by the relevant hostname in packet dumps.

1.2.1.1 Snort Alert Log

```
[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:40:55.510000 158.152.26.39:55440 -> weasel:3128
TCP TTL:41 TOS:0x0 ID:59984 IpLen:20 DgmLen:44
*****S* Seq: 0xF35D32EB Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:40:55.530000 158.152.26.39:55440 -> weasel:8080
TCP TTL:41 TOS:0x0 ID:19637 IpLen:20 DgmLen:44
*****S* Seq: 0xF35D32EB Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [100:2:1] spp_portscan: portscan status from 158.152.26.39: 8 connections across
2 hosts: TCP (8), UDP (0) [**]
08/01-17:45:26.199604

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:40:57.300000 158.152.26.39:55441 -> weasel:3128
TCP TTL:41 TOS:0x0 ID:32099 IpLen:20 DgmLen:44
*****S* Seq: 0x2F2FC691 Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:40:57.310000 158.152.26.39:55441 -> weasel:8080
TCP TTL:41 TOS:0x0 ID:8613 IpLen:20 DgmLen:44
*****S* Seq: 0x2F2FC691 Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:40:59.300000 158.152.26.39:55442 -> weasel:3128
TCP TTL:41 TOS:0x0 ID:61351 IpLen:20 DgmLen:44
*****S* Seq: 0xAC3B8FBF Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:40:59.310000 158.152.26.39:55442 -> weasel:8080
TCP TTL:41 TOS:0x0 ID:18009 IpLen:20 DgmLen:44
*****S* Seq: 0xAC3B8FBF Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [100:2:1] spp_portscan: portscan status from 158.152.26.39: 4 connections across
1 hosts: TCP (4), UDP (0) [**]
08/01-17:45:26.205409

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
```

```
08/01-17:41:01.330000 158.152.26.39:55443 -> weasel:8080
TCP TTL:41 TOS:0x0 ID:37213 IpLen:20 DgmLen:44
*****S* Seq: 0xF35D32EB Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:41:01.390000 158.152.26.39:55443 -> weasel:3128
TCP TTL:41 TOS:0x0 ID:63901 IpLen:20 DgmLen:44
*****S* Seq: 0xF35D32EB Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:41:03.310000 158.152.26.39:55444 -> weasel:8080
TCP TTL:41 TOS:0x0 ID:2605 IpLen:20 DgmLen:44
*****S* Seq: 0x2F2FC691 Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:41:03.380000 158.152.26.39:55444 -> weasel:3128
TCP TTL:41 TOS:0x0 ID:64725 IpLen:20 DgmLen:44
*****S* Seq: 0x2F2FC691 Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [100:2:1] spp_portscan: portscan status from 158.152.26.39: 4 connections across
1 hosts: TCP(4), UDP(0) [**]
08/01-17:45:26.224249

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:41:05.330000 158.152.26.39:55445 -> weasel:8080
TCP TTL:41 TOS:0x0 ID:56856 IpLen:20 DgmLen:44
*****S* Seq: 0xAC3B8FBF Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 3]
08/01-17:41:05.390000 158.152.26.39:55445 -> weasel:3128
TCP TTL:41 TOS:0x0 ID:24947 IpLen:20 DgmLen:44
*****S* Seq: 0xAC3B8FBF Ack: 0x0 Win: 0xC00 TcpLen: 24
TCP Options (1) => MSS: 536
```

1.2.1.2 Snort Portscan Log

```
Aug 1 17:40:55 158.152.26.39:55440 -> weasel:8080 SYN *****S*
Aug 1 17:40:55 158.152.26.39:55440 -> weasel:443 SYN *****S*
Aug 1 17:40:55 158.152.26.39:55440 -> weasel:80 SYN *****S*
Aug 1 17:40:59 158.152.26.39:55442 -> weasel:8080 SYN *****S*
Aug 1 17:40:59 158.152.26.39:55442 -> weasel:443 SYN *****S*
Aug 1 17:41:01 158.152.26.39:55443 -> weasel:80 SYN *****S*
Aug 1 17:40:59 158.152.26.39:55442 -> weasel:3128 SYN *****S*
Aug 1 17:41:03 158.152.26.39:55444 -> weasel:443 SYN *****S*
Aug 1 17:41:03 158.152.26.39:55444 -> weasel:8080 SYN *****S*
Aug 1 17:41:03 158.152.26.39:55444 -> weasel:3128 SYN *****S*
Aug 1 17:41:05 158.152.26.39:55445 -> weasel:80 SYN *****S*
```

1.2.1.3 Tcpdump “full fidelity trace”

```
17:40:55.510000 P 158.152.26.39.55440 > weasel.squid: S 4082971371:4082971371(0) win
3072 <mss 536> (ttl 41, id 59984)
    4500 002c ea50 0000 2906 eacd 9e98 1a27
    .... d890 0c38 f35d 32eb 0000 0000
    6002 0c00 c802 0000 0204 0218 0000
17:40:55.530000 P 158.152.26.39.55440 > weasel.webcache: S 4082971371:4082971371(0)
win 3072 <mss 536> (ttl 41, id 19637)
    4500 002c 4cb5 0000 2906 8869 9e98 1a27
    .... d890 1f90 f35d 32eb 0000 0000
    6002 0c00 b4aa 0000 0204 0218 0000
17:40:55.530000 P 158.152.26.39.55440 > weasel.https: S 4082971371:4082971371(0) win
3072 <mss 536> (ttl 41, id 58435)
    4500 002c e443 0000 2906 f0da 9e98 1a27
    .... d890 01bb f35d 32eb 0000 0000
    6002 0c00 d27f 0000 0204 0218 0000
```

```
17:40:55.530000 P 158.152.26.39.55440 > weasel.http: S 4082971371:4082971371(0) win
3072 <mss 536> (ttl 41, id 43646)
    4500 002c aa7e 0000 2906 2aa0 9e98 1a27
    .... d890 0050 f35d 32eb 0000 0000
    6002 0c00 d3ea 0000 0204 0218 0000
17:40:57.300000 P 158.152.26.39.55441 > weasel.squid: S 791660177:791660177(0) win
3072 <mss 536> (ttl 41, id 32099)
    4500 002c 7d63 0000 2906 57bb 9e98 1a27
    .... d891 0c38 2f2f c691 0000 0000
    6002 0c00 f889 0000 0204 0218 0000
17:40:57.310000 P 158.152.26.39.55441 > weasel.webcache: S 791660177:791660177(0) win
3072 <mss 536> (ttl 41, id 8613)
    4500 002c 21a5 0000 2906 b379 9e98 1a27
    .... d891 1f90 2f2f c691 0000 0000
    6002 0c00 e531 0000 0204 0218 0000
17:40:57.320000 P 158.152.26.39.55441 > weasel.https: S 791660177:791660177(0) win
3072 <mss 536> (ttl 41, id 36641)
    4500 002c 8f21 0000 2906 45fd 9e98 1a27
    .... d891 01bb 2f2f c691 0000 0000
    6002 0c00 0307 0000 0204 0218 0000
17:40:57.370000 P 158.152.26.39.55441 > weasel.http: S 791660177:791660177(0) win 3072
<mss 536> (ttl 41, id 30087)
    4500 002c 7587 0000 2906 5f97 9e98 1a27
    .... d891 0050 2f2f c691 0000 0000
    6002 0c00 0472 0000 0204 0218 0000
17:40:59.300000 P 158.152.26.39.55442 > weasel.squid: S 2889584575:2889584575(0) win
3072 <mss 536> (ttl 41, id 61351)
    4500 002c efa7 0000 2906 e576 9e98 1a27
    .... d892 0c38 ac3b 8fbf 0000 0000
    6002 0c00 b24e 0000 0204 0218 0000
17:40:59.310000 P 158.152.26.39.55442 > weasel.webcache: S 2889584575:2889584575(0)
win 3072 <mss 536> (ttl 41, id 18009)
    4500 002c 4659 0000 2906 8ec5 9e98 1a27
    .... d892 1f90 ac3b 8fbf 0000 0000
    6002 0c00 9ef6 0000 0204 0218 0000
17:40:59.310000 P 158.152.26.39.55442 > weasel.https: S 2889584575:2889584575(0) win
3072 <mss 536> (ttl 41, id 63363)
    4500 002c f783 0000 2906 dd9a 9e98 1a27
    .... d892 01bb ac3b 8fbf 0000 0000
    6002 0c00 bccb 0000 0204 0218 0000
17:40:59.370000 P 158.152.26.39.55442 > weasel.http: S 2889584575:2889584575(0) win
3072 <mss 536> (ttl 41, id 26845)
    4500 002c 68dd 0000 2906 6c41 9e98 1a27
    .... d892 0050 ac3b 8fbf 0000 0000
    6002 0c00 be36 0000 0204 0218 0000
17:41:01.310000 P 158.152.26.39.55443 > weasel.http: S 4082971371:4082971371(0) win
3072 <mss 536> (ttl 41, id 5877)
    4500 002c 16f5 0000 2906 be29 9e98 1a27
    .... d893 0050 f35d 32eb 0000 0000
    6002 0c00 d3e7 0000 0204 0218 0000
17:41:01.320000 P 158.152.26.39.55443 > weasel.https: S 4082971371:4082971371(0) win
3072 <mss 536> (ttl 41, id 16096)
    4500 002c 3ee0 0000 2906 963e 9e98 1a27
    .... d893 01bb f35d 32eb 0000 0000
    6002 0c00 d27c 0000 0204 0218 0000
17:41:01.330000 P 158.152.26.39.55443 > weasel.webcache: S 4082971371:4082971371(0)
win 3072 <mss 536> (ttl 41, id 37213)
    4500 002c 915d 0000 2906 43c1 9e98 1a27
    .... d893 1f90 f35d 32eb 0000 0000
    6002 0c00 b4a7 0000 0204 0218 0000
17:41:01.390000 P 158.152.26.39.55443 > weasel.squid: S 4082971371:4082971371(0) win
3072 <mss 536> (ttl 41, id 63901)
    4500 002c f99d 0000 2906 db80 9e98 1a27
    .... d893 0c38 f35d 32eb 0000 0000
    6002 0c00 c7ff 0000 0204 0218 0000
17:41:03.300000 P 158.152.26.39.55444 > weasel.http: S 791660177:791660177(0) win 3072
<mss 536> (ttl 41, id 20378)
    4500 002c 4f9a 0000 2906 8584 9e98 1a27
    .... d894 0050 2f2f c691 0000 0000
    6002 0c00 046f 0000 0204 0218 0000
17:41:03.310000 P 158.152.26.39.55444 > weasel.https: S 791660177:791660177(0) win
3072 <mss 536> (ttl 41, id 37743)
    4500 002c 936f 0000 2906 41af 9e98 1a27
    .... d894 01bb 2f2f c691 0000 0000
    6002 0c00 0304 0000 0204 0218 0000
17:41:03.310000 P 158.152.26.39.55444 > weasel.webcache: S 791660177:791660177(0) win
3072 <mss 536> (ttl 41, id 2605)
```

```

4500 002c 0a2d 0000 2906 caf1 9e98 1a27
.... .... d894 1f90 2f2f c691 0000 0000
6002 0c00 e52e 0000 0204 0218 0000
17:41:03.380000 P 158.152.26.39.55444 > weasel.squid: S 791660177:791660177(0) win
3072 <mss 536> (ttl 41, id 64725)
4500 002c fcd5 0000 2906 d848 9e98 1a27
.... .... d894 0c38 2f2f c691 0000 0000
6002 0c00 f886 0000 0204 0218 0000
17:41:05.310000 P 158.152.26.39.55445 > weasel.http: S 2889584575:2889584575(0) win
3072 <mss 536> (ttl 41, id 38784)
4500 002c 9780 0000 2906 3d9e 9e98 1a27
.... .... d895 0050 ac3b 8fbf 0000 0000
6002 0c00 be33 0000 0204 0218 0000
17:41:05.320000 P 158.152.26.39.55445 > weasel.https: S 2889584575:2889584575(0) win
3072 <mss 536> (ttl 41, id 40893)
4500 002c 9fbd 0000 2906 3561 9e98 1a27
.... .... d895 01bb ac3b 8fbf 0000 0000
6002 0c00 bcc8 0000 0204 0218 0000
17:41:05.330000 P 158.152.26.39.55445 > weasel.webcache: S 2889584575:2889584575(0)
win 3072 <mss 536> (ttl 41, id 56856)
4500 002c de18 0000 2906 f705 9e98 1a27
.... .... d895 1f90 ac3b 8fbf 0000 0000
6002 0c00 9ef3 0000 0204 0218 0000
17:41:05.390000 P 158.152.26.39.55445 > weasel.squid: S 2889584575:2889584575(0) win
3072 <mss 536> (ttl 41, id 24947)
4500 002c 6173 0000 2906 73ab 9e98 1a27
.... .... d895 0c38 ac3b 8fbf 0000 0000
6002 0c00 b24b 0000 0204 0218 0000

```

1.2.2 Location Of Trace

Home internet connection. See Detect #1 for further details.

1.2.3 Source Of Trace

As Detect #1.

1.2.4 Likelihood Of Address Spoofing?

Low. This attack is directed at HTTP servers and so requires that a valid TCP 3 -way handshake be completed.

1.2.5 Description Of Attack

This attack is searching for open ports usually associated with Web services. Common web oriented ports are tested using SYN packets for a response.

1.2.6 Attack Mechanism

This does not class as an attack itself, but certainly counts as reconnaissance. All the ports scanned offer web services in one form or another (either as end -servers or as proxies). I would not be surprised to find HTTP specific probes directed at the servers that did actually respond at some stage in the (near) future - though to this date this has not occurred.

1. Of the servers which are running standard port 80 bound web -servers, those that did respond with the expected ACK never received the final stage of the handshake, prompting the belief that this a form of “half -open” SYN scan.
2. The ports targeted make the scan look like a RingZero probe, except for the addition of TCP 443 (and the fact that it’s a really old scan).
3. There is evidence of TCP’s typical retry behavior (identical sequence numbers, but differing fragment ID’s)

4. The ISN's are widely spaced, which could indicate a very busy source machine, but I would have thought that such a busy machine would be likely to space subsequent scans more than the timestamps tend to illustrate.
5. Using the document "RPC and NMAP Patterns" from SANS highlights a number of points that are suggestive of an NMAP scan – namely;
 - Very random IPID's
 - A fixed window size of 3072 (one of the candidate sizes)
 - TTL fits within expected range of 37 -59

See <http://www.sans.org/newlook/resources/IDFAQ/NMAP.htm>

6. MSS values are quite small, perhaps indicating a dialup connection (on checking the source address is found to belong to Demon Internet, a large UK based dial-up ISP)

1.2.7 Correlations

If these scans actually are of the "half open" type, then you would not expect to see entries in the webserver logs. None of the systems are running HIDS software, so that rules out another avenue of correlation.

For systems inside the firewall, its log does offer confirmation of the Snort logs. Please note obscured destination addresses.

No.	Date	Time	Action	Service	Source	Destination	Prio.	Rule	S_Port
8384	1 Aug 2001	16:42:10	drop	http	10.0.0.0	...	tcp	20	55141
8385	1 Aug 2001	16:42:10	drop	http	10.0.0.0	...	tcp	20	55141
8390	1 Aug 2001	16:42:10	drop	ftp	10.0.0.0	...	tcp	20	55142
8391	1 Aug 2001	16:42:10	drop	http-proxy	10.0.0.0	...	tcp	20	55142
8392	1 Aug 2001	16:42:10	drop	http	10.0.0.0	...	tcp	20	55142
8393	1 Aug 2001	16:42:10	drop	http	10.0.0.0	...	tcp	20	55142
8394	1 Aug 2001	16:42:12	drop	ftp	10.0.0.0	...	tcp	20	55143
8395	1 Aug 2001	16:42:12	drop	ftp	10.0.0.0	...	tcp	20	55143
8396	1 Aug 2001	16:42:12	drop	http-proxy	10.0.0.0	...	tcp	20	55143
8397	1 Aug 2001	16:42:12	drop	ftp	10.0.0.0	...	tcp	20	55143
8398	1 Aug 2001	16:42:14	drop	http	10.0.0.0	...	tcp	20	55144
8399	1 Aug 2001	16:42:14	drop	http-proxy	10.0.0.0	...	tcp	20	55144
8381	1 Aug 2001	16:42:14	drop	ftp	10.0.0.0	...	tcp	20	55144
8382	1 Aug 2001	16:42:14	drop	http	10.0.0.0	...	tcp	20	55144
8383	1 Aug 2001	16:42:14	drop	http-proxy	10.0.0.0	...	tcp	20	55144
8384	1 Aug 2001	16:42:16	drop	http-proxy	10.0.0.0	...	tcp	20	55145
8385	1 Aug 2001	16:42:16	drop	ftp	10.0.0.0	...	tcp	20	55145
8386	1 Aug 2001	16:42:16	drop	ftp	10.0.0.0	...	tcp	20	55145

Figure 2 - Firewall log showing dropped web service connections

1.2.8 Evidence Of Active Targeting

Low. The services being targeted are generic web services. The portscan covers the whole of the /28 address space, and does not discriminate between servers. The timestamps however suggest that this was specifically targeted at this address range

(or more likely a larger address range than just my /28 portion) – in as much as they are in close succession rather than distributed.

1.2.9 Severity

Criticality	4	Scan is searching for webservers, some of which do exist in the scanned address space.
+ Lethality	2	Not lethal in this context, the results of what seems to be an intelligence gathering phase maybe though.
-		
System Countermeasures	4	None of the systems are running services on the 3128 or 8080 ports. All systems save for “badger” (the honeypot) are modern and running the recommended patches.
+ Network Countermeasures	4	Again with the exception of the honeypot system, the firewall blocked access to the 3128, 8080 and where appropriate 443 ports.
		Port 80 has to be allowed through to support the primary function of the webserver.
Total	-2	Low. Though further monitoring strongly recommended.

1.2.10 Defensive Recommendations

- Recent events continue to illustrate that the correct patches should be installed in a timely fashion.
- Monitor firewall logs for further connection attempts to the ports 3128 and 8080.

1.2.11 Test Question

In the TCPDump capture above, what switches are likely to have been used (choose all that apply)

- a) -a
- b) -q
- c) -x
- d) -vv

Answer

C and D. -x hex dumps packet contents up to SNAPLEN bytes. -vv provides additional packet detail (TTL, IPID).

-a also dumps packet contents, but in ASCII, not hex.
-q - "Quiet" mode.

See man tcpdump for further details.

1.3.4 Description Of Attack

From http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids286&view=research
quoting from www.securityfocus.com

Washington University ftp daemon (wu -ftpd) is a very popular unix ftp server shipped with many distributions of Linux. Wu -ftpd is vulnerable to a very serious remote attack in the SITE EXEC implementation. Because of user input going directly into a format string for a *printf function, it is possible to overwrite important data, such as a return address, on the stack. When this is accomplished, the function can jump into shellcode pointed to by the overwritten eip and execute arbitrary commands as root. While exploited in a manner similar to a buffer overflow, it is actually an input validation problem. Anonymous ftp is exploitable making it even more serious as attacks can come anonymously from anywhere on the internet.

1.3.5 Attack Mechanism

In this instance, it does not seem as though the attack was successful. The documentation that I have been able to locate regarding this does not make any mention of 2.6.0 on RH5.2 being vulnerable. This may be the reason, or it might be that the exploit was not executed correctly.

Whatever the reason, subsequent packet dumps of the activity from “badger” do not reveal any indication of outgoing activity as a result of compromise.

1.3.6 Correlations

This system has been placed outside of the firewall, so local firewall logs are not available. What application level logs for the host exist are included above in “Trace Data”

http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids286&view=research

1.3.7 Evidence Of Active Targeting

Difficult to say. The targeted server is a Linux box running a version of wu -ftpd, however there is nothing further in either the snort logs relating to this source address, or the full tcpdump traces that are also recorded.

1.3.8 Severity

Criticality	4	Exploit targets a seemingly vulnerable FTP server exposed with an anonymous account.
+ Lethality	4	Successful execution of this exploit leads to root on the target system.
-		
System Countermeasures	1	Unfirewalled host, running targeted version of FTP demon.
+ Network Countermeasures	2	The attack was detected.
Total	5	High.

1.3.9 Defensive Recommendations

Upgrade to 2.6.1 as this contains fixes to address this issue.

The file <http://www.wu-ftp.org/CHANGES> includes the following brief description

Changes in 2.6.1: Released 2 Jul, 2000

Fix security leaks that could result in a root shell compromise.

1.3.10 Test Question

Using the data in the Snort Alert Log, which Operating System may the attacker be using? Note that there may not be a correct answer to this one.

- a) Windows 98
- b) Windows NT
- c) Solaris
- d) Linux

Answer

c. Basing the answer on the value of the TTL field. The majority of operating systems tend to use TTL's in the range 64 -128. Solaris is known to use 255, though this can be changed (as it can for other systems).

http://www.switch.ch/docs/ttl_default.html

1.4 Detect #4 – Bulk CGI vulnerability scan

1.4.1 Trace Data

1.4.1.1 Apache Webserver Logs

A portion of the webserver log is shown below.

Log entries have fields with the following meanings - Source IP Address, Time/Date Stamp, Document Requested, HTTP Version, Server Status Code, Size of Document Returned

```
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/flexform.cgi HTTP/1.0"
404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/flexform HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/LWGate HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/lwgate HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/LWGate.cgi HTTP/1.0" 404
0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/lwgate.cgi HTTP/1.0" 404
0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-win/ HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/pu3.pl HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/meta.pl HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/day5datacopier.cgi
HTTP/1.0" 404 0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/webutils.pl HTTP/1.0" 404
0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/tigvote.cgi HTTP/1.0" 404
0
24.23.19.211 -- [21/Jul/2001:17:32:45 +0000] "HEAD /cgi-bin/tpgnrock HTTP/1.0" 404 0
```

```
24.23.19.211 - - [21/Jul/2001:17:32:46 +0000] "HEAD /cgi-bin/webwho.pl HTTP/1.0" 404 0
24.23.19.211 - - [21/Jul/2001:17:32:46 +0000] "HEAD /cgi-bin/form.cgi HTTP/1.0" 404 0
24.23.19.211 - - [21/Jul/2001:17:32:46 +0000] "HEAD /cgi-bin/message.cgi HTTP/1.0" 404
0
24.23.19.211 - - [21/Jul/2001:17:32:46 +0000] "HEAD /cgi-
bin/.cobalt/siteUserMod/siteUserMod.cgi HTTP/1.0" 404 0
24.23.19.211 - - [21/Jul/2001:17:32:46 +0000] "HEAD /cgi-bin/.fhp HTTP/1.0" 404 0
24.23.19.211 - - [21/Jul/2001:17:32:46 +0000] "HEAD /cgi-bin/htsearch HTTP/1.0" 404 0
```

Simple scripts look for failed document requests (characterised by a 4xx or 5xx result code) – included in the appendix.

1.4.2 Location Of Trace

Home internet connection. See Detect #1 for further details.

1.4.3 Likelihood Of Address Spoofing?

Low. This attack is directed at HTTP servers and so requires that a valid TCP 3 -way handshake be completed.

1.4.4 Description Of Attack

Many home grown and commercially available CGI scripts contain exploitable weaknesses - most commonly of the information leak variety (such as reading files outside of the CGI directory). There is no one common footprint to the vulnerabilities.

1.4.5 Attack Mechanism

This is a bulk probe for many different CGI weaknesses. At a guess, given the exclusive use of HEAD methods (rather than GET's probably as a performance enhancement), and the large number of different probes registered, I'd say this was probably Whisker (<http://www.wiretrip.net/~rfp>), but if anyone thinks differently, I'm open to suggestions.

1.4.6 Correlations

Snort detected many of these CGI probes and , when combined with SnortSnarf, provides useful overview data.

Snortsnarf (<http://www.silicondefence.com/snortsnarf>) takes data recorded by Snort and produces a browser based report. In the example below, I have “drilled down” to the full detail the activities of one particular source address – that identified by the Apache scripts.

Snort includes many cross-references to well known vulnerability sites including <http://whitehats.com>, <http://mitre.cve.org> and <http://www.securityfocus.com>. A number of the probes intercepted have entries on one or more of the above sites.

This simplifies the task of identifying previous incidents of these CGI attacks.



Figure 3 - Snortsnarf Summary

1.4.7 Evidence Of Active Targeting

Some. This is an attempt to identify flawed CGI programs, and it was directed at a webserver.

1.4.8 Severity

Criticality	4	Webserver
+ Lethality	1	Server does not run any CGI programs.
-		
- System Countermeasures	4	Server does not run any CGI programs.
+ Network Countermeasures	2	The attack was detected. Even if the host had been firewalled, the attack probes are across HTTP which would have been allowed through.
Total	-1	Low.

1.4.9 Defensive Recommendations

- Ensure that any affected CGI programs are replaced with patched versions as soon as is possible.
- Implement a restrictive firewall policy for outbound traffic as well as inbound. This probably won't stop an HTTP/CGI compromise, but might help reduce the damage that your compromised system can do to other victims. This would be particularly useful if it is stateful as this would probably catch things such as a hostile process bound to port 80 (such as netcat), where a simple packet filter would not.

1.4.10 Test Question

Identify the false statement based on the logs above.

- a) HEADs are used to retrieve only the first portion of the requested document.
- b) HEADs test for the existence of the requested document.
- c) The /cgi-bin/webutils.pl script was not found and executed by the server.
- d) HTTP 1.0 has been used as a transport to maximise compatibility.

Answer

a. The HEAD method checks for the existence of the requested document. For a vulnerability scanner this is a sensible choice as it is trying to identify that a script exists, but usually does not attempt the exploit itself.

1.5 Detect #5 – Low Port-Low Port SYNFIN Scan

1.5.1 Trace Data

1.5.1.1 Snort Alert Log

Please note that much of the content of this log segment has been removed, and that the destination addresses have been sanitised.

```

[**] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
08/09-19:33:32.052372 217.60.238.3:23 -> xx.yy.197.178:23
TCP TTL:31 TOS:0x0 ID:39426 IpLen:20 DgmLen:44
*****SF Seq: 0x35FADD46 Ack: 0x29392866 Win: 0x404 TcpLen: 24
TCP Options (1) => MSS: 536
0x0000: 00 00 00 00 00 01 00 20 6F 08 60 C7 08 00 45 00 ..... o.`...E.
0x0010: 00 2C 9A 02 00 00 1F 06 36 A6 D9 3C EE 03 xx yy .....6.<.>1
0x0020: C5 B2 00 17 00 17 35 FA DD 46 29 39 28 66 60 03 .....5..F)9(f`.
0x0030: 04 04 67 8B 00 00 02 04 02 18 00 00 .....g.....

=====

[**] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
08/09-19:33:32.072372 217.60.238.3:23 -> xx.yy.197.179:23
TCP TTL:31 TOS:0x0 ID:39426 IpLen:20 DgmLen:44
*****SF Seq: 0x35FADD46 Ack: 0x29392866 Win: 0x404 TcpLen: 24
TCP Options (1) => MSS: 536
0x0000: 00 00 00 00 00 01 00 20 6F 08 60 C7 08 00 45 00 ..... o.`...E.
0x0010: 00 2C 9A 02 00 00 1F 06 36 A5 D9 3C EE 03 xx yy .....6.<.>1
0x0020: C5 B3 00 17 00 17 35 FA DD 46 29 39 28 66 60 03 .....5..F)9(f`.
0x0030: 04 04 67 8A 00 00 02 04 02 18 00 00 .....g.....

=====

[**] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
08/09-19:33:32.232372 217.60.238.3:23 -> xx.yy.197.187:23
TCP TTL:31 TOS:0x0 ID:39426 IpLen:20 DgmLen:44
*****SF Seq: 0x638CECE1 Ack: 0x1DB99E53 Win: 0x404 TcpLen: 24
TCP Options (1) => MSS: 536
0x0000: 00 00 00 00 00 01 00 20 6F 08 60 C7 08 00 45 00 ..... o.`...E.
0x0010: 00 2C 9A 02 00 00 1F 06 36 9D D9 3C EE 03 xx yy .....6.<.>1
0x0020: C5 BB 00 17 00 17 63 8C EC E1 1D B9 9E 53 60 03 .....c.....S`.
0x0030: 04 04 BF E7 00 00 02 04 02 18 00 00 .....

=====

[**] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
08/09-19:33:32.252372 217.60.238.3:23 -> xx.yy.197.188:23
TCP TTL:31 TOS:0x0 ID:39426 IpLen:20 DgmLen:44
*****SF Seq: 0x638CECE1 Ack: 0x1DB99E53 Win: 0x404 TcpLen: 24
TCP Options (1) => MSS: 536
0x0000: 00 00 00 00 00 01 00 20 6F 08 60 C7 08 00 45 00 ..... o.`...E.
0x0010: 00 2C 9A 02 00 00 1F 06 36 9C D9 3C EE 03 xx yy .....6.<.>1
0x0020: C5 BC 00 17 00 17 63 8C EC E1 1D B9 9E 53 60 03 .....c.....S`.

```

```
0x0030: 04 04 BF E6 00 00 02 04 02 18 00 00 .....  
+++++  
[**] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]  
08/09-19:33:32.292372 217.60.238.3:23 -> xx.yy.197.190:23  
TCP TTL:31 TOS:0x0 ID:39426 IpLen:20 DgmLen:44  
*****SF Seq: 0x638CECE1 Ack: 0x1DB99E53 Win: 0x404 TcpLen: 24  
TCP Options (1) => MSS: 536  
0x0000: 00 00 00 00 00 01 00 20 6F 08 60 C7 08 00 45 00 ..... o.`...E.  
0x0010: 00 2C 9A 02 00 00 1F 06 36 9A D9 3C EE 03 xx yy .....6.<...>1  
0x0020: C5 BE 00 17 00 17 63 8C EC E1 1D B9 9E 53 60 03 .....c.....S`.  
0x0030: 04 04 BF E4 00 00 02 04 02 18 00 00 .....  
+++++
```

1.5.2 Location Of Trace

Home internet connection. See Detect #1 for further details.

1.5.3 Likelihood Of Address Spoofing?

Low. This probe makes use of an invalid TCP flag combination, however it is still only useful if the attacker can see the results of his probe, suggesting valid source addresses.

1.5.4 Description Of Attack

So-called "stealthy" port scan.

1.5.5 Attack Mechanism

Not an attack as such, but a probe with some significant characteristics.

- The TCP flag combination of SYN and FIN is illegal.
- Low port to Low Port. This is unusual enough to warrant comment, particularly with Telnet where one end would almost¹ always be ephemeral. The source port is likely chosen to penetrate stateless packet filters.
- IPID's remain constant (39426) throughout scan.

Minor points;

- The TCP MSS is 536 bytes, a common length for dialup (or typically PPP) connections, but all the connection attempts happen within one second perhaps suggesting a higher-speed connection.
- SYN packets correctly do not contain any payload data.

1.5.6 Correlations

Packets with an identical signature (SF, static IPID, fixed Window Size) reported at SANS.

<http://www.sans.org/y2k/111600.htm>

¹ I've never yet seen a stock telnet client connection that doesn't initiate off of an ephemeral port, but that's not to say there isn't one somewhere.

It is also possible that this is a form of OS determination. Older versions of Linux are known to have erroneously responded to SYN-FIN packets with SYN-FIN-ACK, though it does not exhibit NMAP's OS identification characteristics.

<http://archives.neohapsis.com/archives/snort/2000-07/0183.html>

1.5.7 Evidence Of Active Targeting

Low. This scan covers the entire /28 address range.

1.5.8 Severity

Criticality	2	General scan, not highly targeted.
+ Lethality	1	None of the machines scanned expose port 23.
-		
- System Countermeasures	4	None of the machines scanned expose port 23.
+ Network Countermeasures	4	Firewall actively blocks incoming telnet connections.
Total	-5	Low.

1.5.9 Defensive Recommendations

Nothing significant. None of the machines offer telnet as a service and the firewall blocks incoming connections anyway.

1.5.10 Test Question

What are the TCP options used and in byte terms, how are they defined?

- 2 bytes Window Size, 2 Bytes MSS
- 1 Byte MSS, 1 byte NOP
- 1 Byte Kind, 1 Byte Length, 2 Bytes MSS
- 2 Byte Selective ACK, 2 Bytes MSS.

Answer

C. RFC 793 defines each TCP option as follows;

Field	Length	Purpose
Kind	1 Byte	Option Identifier
Length (Bytes)	1 Byte	Total length of options (including Kind/Length fields)
Option Data	Variable	Variable

The MSS field is defined as Kind=2, Length=4, MSS Value=2 bytes. In the event that options don't reach a multiple of 4 bytes in length, they are padded with NOP's.

2 Assignment #2 - What happens before and after your sensor alarms?

2.1 Introduction

One of the problems with Intrusion Detection sensors is that they are reactive. Providing alerts based on a packet signature is a valuable ability, but sensors will only alert on what they know about.

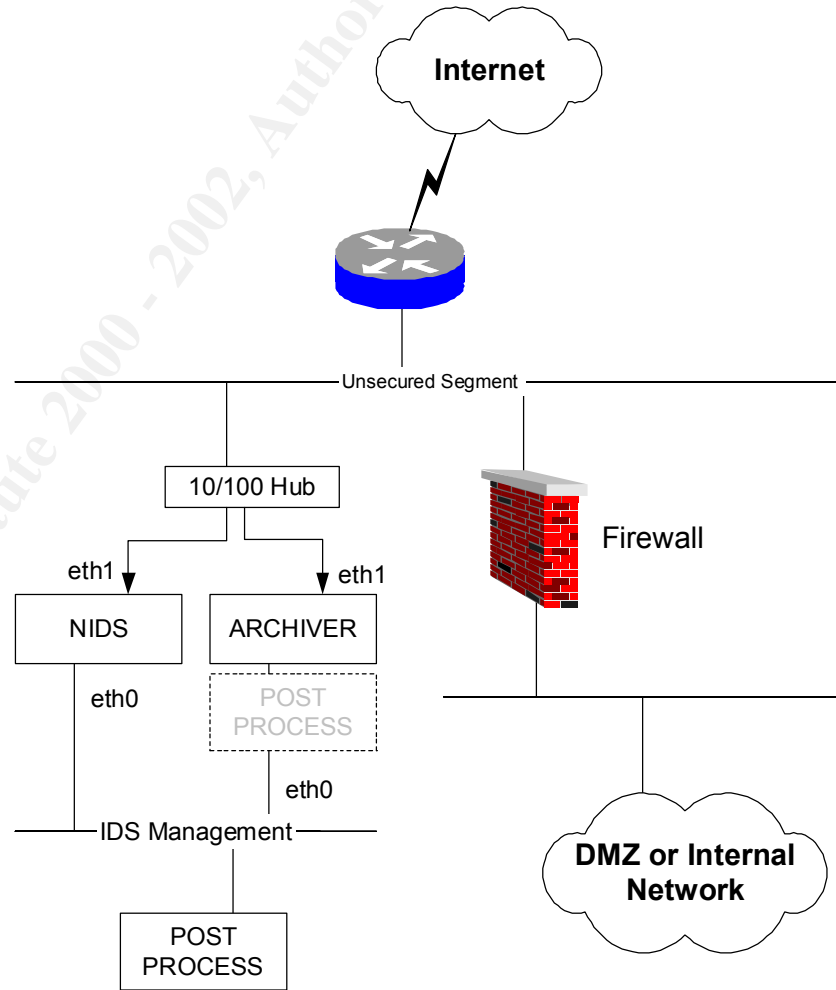
With today's fast processors, cheap RAM, and huge hard disks (100GB IDE units at sub-200UKP) the ability to provide "full fidelity" storage even on relatively fast links becomes a possibility.

An approach is outlined below that will enable answers to the question "what happened before and after a given alert", but basically rely on a machine termed an "Archiver" that records off the wire everything that the sensor is exposed to. The idea is that the data obtained from the Archiver is post processed to provide more detail than the sensor can provide alone – critically covering the periods before and after the exploit was detected.

2.2 Method

In order for network based intrusion detection to work, you must provide the sensor with a data stream for it to monitor. This may be done either by providing the infrastructure on a shared media (a hub), when using Cisco equipment via the SPAN mechanism that allows frames from one or more ports to be copied to another port for analysis, or through the use of so-called "Taps".

The diagram below does not make any distinction between the methods used to provide the infrastructure supporting the "unsecured segment". It does however rely on the use of a shared media to feed the Sensor and the Archiver. If you are using a Cisco SPAN port, span the incoming feed (the Ethernet interface of your internet router), and copy the frames to the switch port linked to your hub.



2.3 What Do you Need?

2.3.1 The Hub

The use of a hub has two main advantages. Firstly, these days they're cheap – a quick scout around the web finds 5 port 10/100 hubs for just over \$40. Now you might want to pay a little more than that for something with an integrated PSU, or one that's rack mountable, but the point here is that they're inexpensive and will do the job just fine.

Secondly the nature of the hub, is “what one port sees, all ports see” - that is, the collision domain typically covers all of the ports on the unit. When you want to work independently on multiple copies of the same frame, that's a real advantage.

You'll need to attach the hub to the switch supporting the "unsecured segment" in the diagram. Usually this requires a crossed Ethernet cable instead of the more usual straight ones. Many hubs have a dedicated or switchable uplink port that can be used with a standard Ethernet patch lead.

Potentially, a 100mb hub installed on a transit network (i.e. one with no hosts), should be able to support relatively high traffic rates – almost certainly capable of the majority of links here in Europe where bandwidth is still very expensive. Adding additional hosts onto the transit network will increase the risk of collisions and decrease throughput.

2.3.2 The Archiver

The Archiver records raw packets from the wire to disk, and ideally, should do nothing else. If you have the hardware available, perform post-processing elsewhere. This is not essential, but performing CPU intensive operations does run the risk of causing dropped packets. I don't have this luxury.

This machine can be running any operating system capable of supporting packet capture software, but I've opted for a Linux version of tcpdump. tcpdump is built on the libpcap system and is as close to a UNIX standard as you'll get. There are a variety of tools that can read tcpdump format files to help with the analysis procedure (including Snort). Examples of these are TCPSLICE and TCPTRACE.

As far as hardware configuration goes the Archiver is heavy on disk space (though this can be tuned as discussed below), but CPU and RAM levels are not so critical. For sub \$1000 you'll get a quick PC (~1 GHz), with adequate memory (256Mb), and big disk (20-60 GB).

Having multiple (or at least dual) network cards is preferred as it allows the listening interface to be passive. High spec. graphics cards are an unnecessary extravagance.

My Archiver is a 1Ghz, 256Mb, 20GB, Dual NIC, Pentium 3 running Redhat Linux 7.1

You'd probably want to apply the same hardening mechanisms to the Archiver as you would to the sensor. A good place to start is with Bastille

You do need to bear in mind Operating System limits on the maximum file sizes. Most Linux implementations (or at least those using ext2 as the file system) have a limit of 2GB per file. Other Operating Systems will vary.

2.3.3 Your Sensor

In my setup Snort is used as a sensor. Snort has an option to read tcpdump files and process these rather than process on-line from an interface, which makes it ideal for post as well as online processing.

If you're using a hub to feed your sensors, and budget allows, multiple systems in parallel from different vendors may be an idea. You'll need to consider how to post-process data from multiple formats though.

2.4 Task List

Below is only a recommendation based on a process that I've found to work quite well.

- Configure the internal interface (eth0 in the diagram) as appropriate for your network.
- Configure the external interface (eth1). Don't bother with an IP address, as recent versions of libpcap don't require an IP stack if used in promiscuous mode.
- Configure access to the Archiver from your internal network. If possible use SSH, particularly if you have had to configure IP on eth1.
- Configure your packet capturing software to start at boot.
- Run a scheduled job to rotate the capture files at a frequency suitable for your network.
- Move the archive files elsewhere to post-process if applicable. There are many ways to do this, SCP, FTP, HTTP (perhaps over SSL). You may also want to look at RSYNC as it provides a number of useful features such as compression and SSH support.
- Post Process as appropriate. A good approach (with Snort) is to replay the captured tcpdump file with the `-r` option. If you do this multiple times with different configuration files, you can achieve activity snapshots for different types of traffic rather than a single jumbled up alert file.

2.5 Tuning

To a degree, tuning is a personal or site specific preference and I'm just including a couple of things here that you might want to consider.

2.5.1 Packet Sizes

If you have the disk space it may be worth capturing the entire packet payload for future analysis. However, you might decide that all you really need is the first 384 (or some other value) bytes of a given packet.

Quite how you arrive at an appropriate figure is up to you, but there's a clear relationship between the size of the packets you capture and length of history that you maintain.

2.5.2 Encrypted Traffic

If you can't read the traffic, why keep it? Perhaps good candidates for exclusion are HTTPS, SSH sessions (TCP 443 and 22), and the IP protocols that IPSEC uses (IP Protocol types 50 & 51, UDP 500)

2.5.3 Bulk Traffic

You could argue that bulk traffic is the least interesting. You might want to record only smaller packets as these may contain interactive content such as telnet or rlogin. File transfers, web sessions and other "bulk" traffic flows generally use larger packets for performance reasons.

2.6 Relevant Scripts

Below are some of the relevant configuration files that I use to implement the system described above.

2.6.1 Script To Start At Boot

My server is a RedHat Linux box, so the script below is suitable for placing as part of /etc/rc.d/init.d to be called at boot. This is just a copy of one of many init scripts amended accordingly and I'm certain that there are more elegant ways of achieving the same goals.

```
#!/bin/bash
# chkconfig: - 50 50
# description: TCP recorder startup
#

# source function library
. /etc/init.d/functions

OPTIONS="-i eth1 -p -s 1514"
RETVAL=0
prog="tcpdump"
LOG_NAME="tcpdump.log"
FNAME=`date +%Y%m%d-%H%M`

start() {
    echo -n $"Starting $prog: "
    tcpdump $OPTIONS -w /usr/tcpdump/$LOG_NAME &
    RETVAL=$?
    echo
    touch /var/lock/subsys/tcpdump
    return $RETVAL
}

stop() {
    echo -n $"Stopping $prog: "
    killproc /usr/tcpdump
    RETVAL=$?
    echo
    rm -f /var/lock/subsys/tcpdump
    mv /usr/tcpdump/tcpdump.tcp /usr/tcpdump/$FNAME.tcp
    return $RETVAL
}

reload(){
    stop
    start
}
```

```

restart(){
    stop
    start
}

condrestart(){
    [ -e /var/lock/subsys/tcpdump ] && restart
    return 0
}

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    restart)
        restart
        ;;
    reload)
        reload
        ;;
    condrestart)
        condrestart
        ;;
    status)
        status tcpdump
        RETVAL=$?
        ;;
    *)
        echo $"Usage: $0 {start|stop|status|restart|condrestart|reload}"
        RETVAL=1
esac
exit $RETVAL

```

2.6.2 Crontab entry to Rotate TCPDUMP files

I don't have enough bandwidth to necessitate swapping archive files intra-day. The 2Gb Linux file size limit is usually sufficient, hence the file is switched just before midnight.

```
59 23 * * * /root/rotate_tcpdump
```

2.6.3 Script to Rotate TCPDUMP files

```

PID=`ps -ef | grep tcpdump | egrep -v grep | awk '{print $2 }'`
FNAME=`date +%Y%m%d-%H%M`

kill $PID
mv /usr/tcpdump/tcpdump.tcp /usr/tcpdump/$FNAME.tcp

# restart tcpdump
/usr/sbin/tcpdump -i eth1 -p -w /usr/tcpdump/tcpdump.tcp -s 1514 &

# snort the file newly created file
# place output files in default /var/log/snort directory
/usr/local/bin/snort -A full -c /etc/snort/snort.conf -X -P 1514 \
-r /usr/tcpdump/$FNAME.tcp

```

2.7 Conclusion

Hopefully this will prove useful. In a scenario like this, there isn't a "right" way of doing things. All manner of software can be combined to provide a set of tools that will give you just a little more data than your sensor can.

Ultimately choices will have to be made about the hardware and software that you use, data transport mechanisms and security, and the methods used to perform any further analysis.

2.8 References

1. Cisco SPAN mechanism. See <http://www.cisco.com/warp/public/473/41.html>
2. TCPDUMP / TCPSLICE. See <http://www.tcpdump.org/related.html>
3. TCPTRACE. See <http://www.tcptrace.org/>
4. SNORT. See <http://www.snort.org/>
5. Bastille hardening scripts. See <http://www.bastille-linux.org/>
6. IP Protocol Types - <http://packetderm.cotse.com/CIE/RFC/1700/3.htm>
7. RYSNC - rsync is an open source utility that provides fast incremental file transfer. See <http://www.rsync.org/>
8. Stevens' TCP/IP Illustrated Vol 1

3 Assignment #3 - Analyse This

3.1 List of Analysed Files

The files analysed for this assignment are shown below. There was no particular reason for choosing these files. Each category of data was combined into a single file for processing.

Scans	Alerts	Out Of Spec
Scans.010621	Alert.010621	oos_Jun.21.2001
Scans.010622	alert.010622	oos_Jun.22.2001
Scans.010623	alert.010623	oos_Jun.23.2001
Scans.010624	alert.010624	oos_Jun.24.2001
Scans.010625	alert.010625	oos_Jun.25.2001
<i>bigscans</i>	<i>bigalerts</i>	<i>bigoos</i>

3.2 Overview

In common with many other educational establishments, ingress and egress filtering does not seem to have previously been a priority, thus opening up internal systems to attack. A brief overview of the top three issues follows.

3.2.1 Multicast Destinations

From the alerts that Snort has generated, it would seem as though the most serious problem is the quantity of packets with dubious source and destination addresses. However, the bulk of these packets are destined for multicast addresses, and so may be legitimate traffic after all.

Further investigation shows that in the raw Snort alert file most of the reported "UDP SRC and DST" alerts relate to multicast destination addresses, and that for the majority of the occurrences the addresses are registered to Yahoo! Broadcast Services, a multi-media content delivery site. With this in mind further analysis is conducted with these sources removed as they are considered misreported. Alert, Source and Destination summaries from the raw data are available in section 4.2

The Snort configuration file should be updated to reflect the relevant multicast destination addresses as belonging to \$HOME_NET.

3.2.2 Trojan Activity

After discounting multicast traffic, the biggest single worry is the seeming prevalence of Sub7 traffic. This is largely a result of the "open door" networking policy, and the implementation of border packet filters (discussed in 3.9.1 below) would put a stop to this and remove 44% (post multicast removal) of the alerts.

3.2.3 Hostile Sources

Sixty-nine source addresses from the block reference by L -ISDNNET-99051 account for 37% of the detected alerts. If you already have reason to believe that this is a hostile source, use a packet filter and block them.

3.3 Prioritised Detects

The table below identifies the ten most frequent Snort detects.

Count	Alert Detail	Sources	Dests.
21600	Possible trojan server activity	2970	9149
17946	Watchlist 000220 IL-ISDNNET-99051	69	19
3612	External RPC call	15	1184
2032	SMB Name Wildcard	194	710
1023	connect to 515 from outside	8	638
698	UDP SRC and DST outside network	31	91
574	Watchlist 000222 NET-NCFC	8	8
321	Queso fingerprint	23	32
296	Back Orifice	3	180
295	High port 65535 tcp - possible Re	15	18

49260	total alerts		
38950	total portscans		

88210	total events		

3.3.1 Possible trojan server activity

Snort has identified some signs of a host compromised by a Trojan application. All of these alerts relate to traffic either to or from TCP port 27374 - the default port for the well-known Sub7 Trojan (as shown in Figure 4)

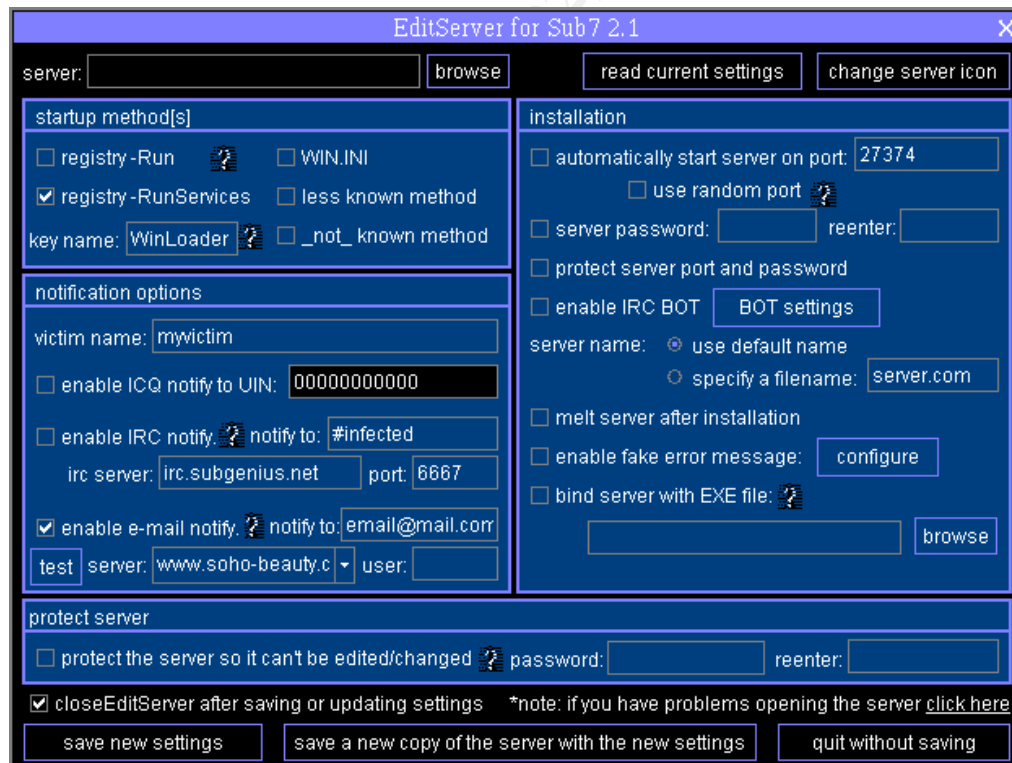


Figure 4 - Sub7 configuration

Recent Snort rules include binary content to help identify real Sub7 activity as opposed to traffic that happens to be using the ephemeral port 27374. Given the quantity of detects here though, Snort's assertion is likely to be correct.

3.3.2 Watchlist 000220 IL-ISDNNET-990517

The network 212.179.0.0 appears to have been singled out for Snort to alert on. The alerts here seem to be based on the originating network rather than signature based.

3.3.3 External RPC call

Machines within MY.NET are attempting to access external RPC services. RPC programs themselves serve legitimate purposes, but there are many well-documented problems with insecure programs.

Without further analysis it is not possible to say what proportion of this traffic is legitimate and what portion is attempting to exploit RPC vulnerabilities.

3.3.4 SMB Name Wildcard

Microsoft Windows systems have a tendency to issue NETBIOS packets to any host that the client machine communicates with, irrespective of the other party's operating system and offered services, so you would expect to see a sizeable number of these packets on any network.

While many of the packets may be legitimate, it is clear following analysis of the logs that there is a large scale NETBIOS scan being conducted too.

3.3.5 connect to 515 from outside

The LPR service that provides printing services to Unix clients and servers resides on port 515. Exploit code is available to provide an attacking user root access via misuse of the LPR daemon.

Unless the university offers printing services to external hosts, these should be classed as a definite intrusion attempt.

3.3.6 UDP SRC and DST outside network

Here Snort has detected packets where neither the source or destination IP addresses are part of the MY.NET address space. Assuming that Snort is correctly configured, it is very likely that machines within MY.NET are sending packets to external addresses with forged (spoofed) source addresses in order to mask their identity.

Spoofed UDP packets form the basis of many DDOS (Distributed Denial Of Service) attacks.

A number of the IP addresses recorded against this alert are invalid (0.14.226.1 for example), or not generally considered routable (RFC1918 – 10.0.1.2 for example).

The majority of the packets recorded here are directed at the ports associated with Windows naming services (137), or DNS (53). A small number are using the UDP ports that are usually associated with BOOTP requests.

3.3.7 Watchlist 000222 NET-NCFC

In the same manner as 3.3.2 above, the network 159.226.0.0 appears to have been singled out for Snort to alert on. Again, the alerts here seem to be based on the originating network rather than any particular signature.

3.3.8 Queso fingerprint

Queso uses "crafted" TCP/IP packets to try and identify a remote Operating System from its IP fingerprint.

The standards that dictate how TCP/IP should be implemented (RFC's) allow for variation in some elements of the IP stack's operation (the order of TCP options for example). These subtle differences act as a fingerprint.

One of the characteristics of Queso is the use of reserved bits/flags in the TCP header. Recently these bits have been adopted by RFC2481 as providing a form of flow-control through a technique known as ECN.

It is possible that this is a valid packet from an operating system utilising an ECN aware IP stack, or it could be a valid Queso detect. If a valid detect, this is not dangerous in itself (though there is a slight risk that a malformed packet will crash a particular host), but is a definite reconnaissance indicator.

3.3.9 Back Orifice

Back Orifice is a well-known Trojan application providing an attacker a number of mechanisms to interfere with compromised machines.

These features include

- Keystroke Logging
- File Transfer
- File Sharing
- Password Dumping
- Remote Command Shell

Machines with Back Orifice installed are frequently used to attack other systems.

3.3.10 High port 65535 tcp - possible Red Worm - traffic

There are at least two Trojans that are known to bind to TCP port 65535 - RC1 and Adore.

Without access to the rule set that generated this alert we can't say exactly what triggered this alarm.

3.4 Top Talkers

3.4.1 Port Scans

Your attention is drawn to the MY.NET devices that appear in the Top Sources list below. These machines may or may not be compromised but are certainly engaged in hostile activities.

3.4.1.1 Top Sources

Count	Source Addresses
20050	139.134.102.192
18451	MY.NET.160.114

```

13485 205.188.244.121
13276 211.72.171.75
12575 165.230.53.35
12368 194.100.55.131
11730 MY.NET.98.167
 9565 205.188.233.121
 7110 205.188.246.121
 6831 205.188.233.153

```

125441

3.4.1.2 Top Destinations

Count Destination Addresses

```

-----
4092 MY.NET.110.33
3677 MY.NET.110.169
3534 MY.NET.145.166
3397 MY.NET.108.13
2986 MY.NET.178.154
2688 MY.NET.108.15
2586 MY.NET.109.62
2277 MY.NET.178.222
2230 MY.NET.106.178
2197 MY.NET.111.30
-----

```

29664

3.4.2 Alerts

The tables below identify the addresses that most frequently occur in the Alert files. A breakdown of the activity for each address is also included.

The tables highlight that the majority of the attackers are single minded in their goal - that is looking for only one vulnerability. This is often an indication that automated tools are being employed in a "fire and forget" manner rather than someone using "cause and effect" to approach a compromise.

3.4.2.1 Top Sources

Top 10 Source Addresses	Alert breakdown for this host
14370 212.179.79.2	14370 Watchlist 000220 IL-ISDNNET-990517
7050 129.170.104.19	7050 Possible trojan server activity
5606 216.220.164.141	5606 Possible trojan server activity
2893 212.179.47.70	2893 Watchlist 000220 IL-ISDNNET-990517
1492 165.230.53.35	1492 SMB Name Wildcard
1176 211.152.241.1	1176 External RPC call
734 24.147.14.159	734 External RPC call
493 159.226.41.166	493 Watchlist 000222 NET-NCFC
425 216.220.167.94	425 Possible trojan server activity
394 24.27.62.134	394 External RPC call

34633 events	

3.4.2.2 Top Destinations

Top 10 Destination Addresses Alert breakdown for this host

```

-----
14379 MY.NET.218.198
                                     14369 Watchlist 000220 IL-ISDNNET-990517
                                     10 WinGate 1080 Attempt
4675 216.220.164.141
                                     4675 Possible trojan server activity
2894 MY.NET.97.175
                                     2893 Watchlist 000220 IL-ISDNNET-990517
                                     1 Null scan!
2738 129.170.104.19
                                     2738 Possible trojan server activity
300 MY.NET.100.56
                                     300 Watchlist 000222 NET-NCFC
257 MY.NET.70.97
                                     149 Watchlist 000220 IL-ISDNNET-990517
                                     64 Queso fingerprint
                                     38 Null scan!
                                     3 High port 65535 tcp - possible Red Worm
                                     2 Possible trojan server activity
                                     1 SYN-FIN scan!
224 MY.NET.218.230
                                     219 Tiny Fragments - Possible Hostile Activity
                                     2 Possible trojan server activity
                                     2 Watchlist 000220 IL-ISDNNET-990517
                                     1 Queso fingerprint
212 MY.NET.97.47
                                     212 Watchlist 000220 IL-ISDNNET-990517
193 MY.NET.100.83
                                     193 Watchlist 000222 NET-NCFC
148 130.132.143.42
                                     148 UDP SRC and DST outside network
-----
26020 events
-----

```

3.4.3 Out Of Spec

Out of Spec packets violate the TCP/IP specifications in some way. This may be due to corruption in transit, badly crafted packets (in correct checksums for example), or just perceived OOS by an analysis tool.

The belief here is that many of the OOS packets have been classified as such because of the existence of ECN data in the reserved bit fields of the TCP options header. More recent versions of Snort (1.8+) correctly identify ECN options.

However, a number of the packets contain either unknown IP or TCP options.

3.4.3.1 Top Sources

```

Top OOS Sources
-----
293 199.183.24.194
226 210.77.146.33
133 193.226.113.248
126 213.116.114.212
125 209.48.248.58
112 111.111.111.111
111 217.0.72.73
90 213.116.160.205
61 192.168.1.1
26 62.243.160.209
-----
1303
-----

```

3.4.3.2 Top Destinations

```

Top OOS Destinations
-----
382 MY.NET.100.165
285 MY.NET.70.97
-----

```

```
180 MY.NET.253.114
133 216.235.163.163
105 MY.NET.253.43
103 MY.NET.253.42
91 MY.NET.253.41
40 216.235.163.151
34 MY.NET.253.125
30 MY.NET.150.225
```

2686

3.5 External Source Registration Details

Included below are the registration details for the source addresses identified in section 3.4.2.1 above.

3.5.1 212.179.79.2

```
inetnum:      212.179.79.0 - 212.179.79.63
netname:      CREOSCITEX
descr:        CREOSCITEX -SIFRA
country:      IL
admin-c:      ZV140 -RIPE
tech-c:       NP469 -RIPE
status:       ASSIGNED PA
notify:       hostmaster@isdn.net.il
mnt-by:       RIPE -NCC-NONE-MNT
changed:      hostmaster@isdn.net.il 20001109
source:       RIPE
```

```
route:        212.179.0.0/17
descr:        ISDN Net Ltd.
origin:       AS8551
notify:       hostmaster@isdn.net.il
mnt-by:       AS8551 -MNT
changed:      hostmaster@isdn.net.il 19 990610
source:       RIPE
```

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

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

3.5.2 129.170.104.19

Dartmouth College ([NET-DART-ETHER](#))
Kiewit Computation Center

Hanover, NH 03755
US

Netname: DART -ETHER
Netblock: [129.170.0.0](#) - [129.170.255.255](#)

Coordinator:
Campbell, Stephen ([SC59-ARIN](#)) steve@AVALON.DARTMOUTH.EDU
603-646-3231

Domain System inverse mapping provided by:

NS1.DARTMOUTH.EDU	129.170.17.4
NS2.DARTMOUTH.ED U	129.170.16.4
MAGGIE.TELCOM.ARIZONA.EDU	128.196.128.233

Record last updated on 21 -Dec-1999.
Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.5.3 216.220.164.141

Pennsylvania Online ([NETBLK-PAONLINE-1](#))
PO Box 6501
Harrisburg, PA 17112
US

Netname: PAONLINE -1
Netblock: [216.220.160.0](#) - [216.220.175.255](#)
Maintainer: PAON

Coordinator:
Peace, George ([GP11-ARIN](#)) george@PAONLINE.NET
(717) 657-0000 (FAX) (717) 657-0132

Domain System inverse mapping provided by:

NS1.PAONLINE.COM	198.69.90.250
NS2.PAONLINE.COM	198.69.90.11
NS3.PAONLINE.COM	207.44.20.1

ADDRESSES WITHIN THIS BLOCK ARE NON -PORTABLE

Record last updated on 27 -Feb-2001.
Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.5.4 165.230.53.35

Rutgers University ([NET-RUTGERS-B2](#))
Telecommunications Division Room 018, Hill Center, Busch Campus
Brett Road
Piscataway, NJ 08855 -0879
US

Netname: RUTGERS -B2
Netblock: [165.230.0.0](#) - [165.230.255.255](#)

Coordinator:

Rutgers University Computing Services (RU-ORG-ARIN)
netmanager@TDMX.RUTGERS.EDU
+1 732 -445-0327

Domain System inverse mapping provided by:

DNS1.RUTGERS.EDU	165.230.144.131
DNS2.RUTGERS.EDU	128.6.21.9
RU-UFL.RUTGERS.EDU	128.227.128.162
TURTLE.MCC.COM	128.62.1.215

Record last updated on 09 -Aug-2000.

Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.5.5 211.152.241.1

Search results for '211.152.241.1'

inetnum	211.152.224.0 - 211.152.255.255
netname	JSBBNET
descr	JS -CASIL BCTV Network Data Co., Ltd. ÄÏ³©ÈÖÏi¹ã²¥µçÈÓíøÃçÊý³áÝÓÐÏ¹«È³¼
country	CN
admin-c	YJ110-AP , inverse
tech-c	DX37-AP , inverse
mnt-by	MAINT-CNNIC-AP , inverse
changed	luoyan@cnnic.net.cn 20010611
source	APNIC

person	Yin Jinjun , inverse
address	9/F, Sujian Building, 31 -1 Yunnan Road,
Nanjing, China	
country	CN
phone	+86 -025-3235063
fax-no	+86 -025-3235066
e-mail	jjyin@public1.ptt.js.cn , inverse
nic-hdl	YJ110-AP , inverse
mnt-by	MAINT-CNNIC-AP , inverse
changed	ipas@cnnic.net.cn 20010131
source	APNIC

person	Ding Xinquan , inverse
address	9/F, Sujian Building, 31 -1 Yunnan Road,
Nanjing, China	
country	CN
phone	+86 -025-3235069
fax-no	+86 -025-3260102
e-mail	js-casil@public1.ptt.js.cn , inverse
nic-hdl	DX37-AP , inverse
mnt-by	MAINT-CNNIC-AP , inverse
changed	ipas@cnnic.net.cn 20010131
source	APNIC

3.5.6 24.147.14.159

MediaOne NorthEast ([NET-M1-NE-2](#))
27 Industrial Ave.
Chelmsford, MA 01 824

US

Netname: M1 -NE-2
Netblock: [24.147.0.0](#) - [24.147.255.255](#)
Maintainer: MDON

Coordinator:
MediaOne NorthEast ([ZM117-ARIN](#)) abuse@mediaone.net
978-244-4020

Domain System inverse mapping provided by:

NS3.MEDIAONE.NET [24.128.1.82](#)
NS4.MEDIAONE.NET [24.130.1.43](#)
NS5.MEDIAONE.NET [24.129.0.103](#)

Record last updated on 10 -Aug-2001.
Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.5.7 159.226.41.166

The Computer Network Center Chinese Academy of Sciences ([NET-NCFC](#))
P.O. Box 2704 -10,
Institute of Computing Technology Chinese Academy of Sciences
Beijing 100080, China
CN

Netname: NCFC
Netblock: [159.226.0.0](#) - [159.226.255.255](#)

Coordinator:
Qian, Haulin ([QH3-ARIN](#)) hlqian@NS.CNC.AC.CN
+86 1 2569960

Domain System inverse mapping provided by:

NS.CNC.AC.CN [159.226.1.1](#)
GINGKO.ICT.AC.CN [159.226.40.1](#)

Record last updated on 25 -Jul-1994.
Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.5.8 216.220.167.94

Pennsylvania Online ([NETBLK-PAONLINE-1](#))
PO Box 6501
Harrisburg, PA 17112
US

Netname: PAONLINE -1
Netblock: [216.220.160.0](#) - [216.220.175.255](#)
Maintainer: PAON

Coordinator:
Peace, George ([GP11-ARIN](#)) george@PAONLINE.NET
(717) 657-0000 (FAX) (717) 657-0132

Domain System inverse mapping provided by:

NS1.PAONLINE.COM [198.69.90.250](#)
NS2.PAONLINE.COM [198.69.90.11](#)
NS3.PAONLINE.COM [207.44.20.1](#)

ADDRESSES WITHIN THIS BLOCK ARE NON -PORTABLE

Record last updated on 27 -Feb-2001.
Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.5.9 24.27.62.134

ServiceCo LLC - Road Runner ([NET-ROAD-RUNNER-1](#))
13241 Woodland Park Road
Herndon, VA 20171
US

Netname: ROAD -RUNNER-1
Netblock: [24.24.0.0](#) - [24.30.95.255](#)
Maintainer: SCRR

Coordinator:
ServiceCo LLC ([ZS30-ARIN](#)) abuse@rr.com
1-703-345-3416

Domain System inverse mapping provided by:

DNS1.RR.COM	24.30.200.3
DNS2.RR.COM	24.30.201.3
DNS3.RR.COM	24.30.199.7
DNS4.RR.COM	65.24.0.172

Record last updated on 15-Aug-2001.
Database last updated on 28 -Aug-2001 23:14:19 EDT.

3.6 Correlations

Correlations for detects listed in section 3.3 follow below.

3.6.1 Possible trojan server activity

Sub7 activity is very common and the Trojan itself has gone through a number of iterations. Reports have been previously submitted to SANS regarding activity by client searching for machines running the Sub7 server component.

<http://www.sans.org/y2k/032700.htm>
<http://www.sans.org/y2k/091900.htm>
http://vil.mcafee.com/dispVirus.asp?virus_k=10171&

Work has been recently carried out by Team2600 to port the Sub7 engine to the Apple Mac platform (both Mac OS X and earlier versions)

3.6.2 Watchlist 000220 IL-ISDNNET-990517

Previous submissions are littered with references to addresses in the 212.179.0.0 network.

http://www.sans.org/y2k/practical/Fred_Portnoy_GCIA.doc
http://www.sans.org/y2k/practical/Stephan_Odak.doc
http://www.sans.org/y2k/practical/Tony_Smith.doc
<http://www.sans.org/y2k/051900.htm>

3.6.3 SMB Name Wildcard

SMB Name Wildcard queries are a common occurrence, and a signature match can be produced using just standard Windows commands (NBTSTAT)

http://www.sans.org/newlook/resources/IDFAQ/port_137.htm

3.6.4 connect to 515 from outside

A number of people have reported this type of connection event previously. Without access to the payload data, we cannot say with certainty that this was an LPR exploit.

<http://www.sans.org/y2k/021401.htm>

<http://www.incidents.org/archives/intrusions/msg01143.html>

<http://www.incidents.org/archives/intrusions/msg01192.html>

3.6.5 UDP SRC and DST outside network

There aren't specific correlations for this signature, but a search on Google gives some indication of their frequency.

<http://www.google.com/search?hl=en&q=%22spoofed+UDP+packets%22>

3.6.6 Watchlist 000222 NET-NCFC

Previous occurrences of detects with sources belonging to the 159.226.0.0 network are contained in the submissions shown below.

http://www.sans.org/y2k/practical/Stephan_Odak.doc

http://www.sans.org/y2k/practical/Tony_Smith.doc

http://www.sans.org/y2k/practical/Sidney_Faber_gcia.doc

3.6.7 Queso fingerprint

Queso is a well known fingerprinting tool (though largely superseded by NM AP). Detailed explanations are available at the links below.

<http://www.whitehats.com/info/IDS29>

<http://www.sans.org/y2k/ecn.htm>

Note particularly that packets from ECN enabled Linux machines are often labelled as Queso attempts. Max Vision points out that you can reduce the number of false positives by looking for this signature in combination with high value TTL's.

3.6.8 Back Orifice

Back Orifice by Cult of the Dead Cow is another "Remote Administration Tool". Previous SANS candidates have reported BO activities.

http://www.sans.org/y2k/practical/Stephan_Odak.doc

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

3.6.9 High port 65535 tcp - possible Red Worm – traffic

<http://archives.neohapsis.com/archives/incidents/2000-04/0050.html>

<http://archives.neohapsis.com/archives/incidents/2000-03/0052.html>

3.7 Out Of Spec (OOS) Link Analysis

The Link Graph below demonstrates the relationship between the most common OOS source address and its destinations.

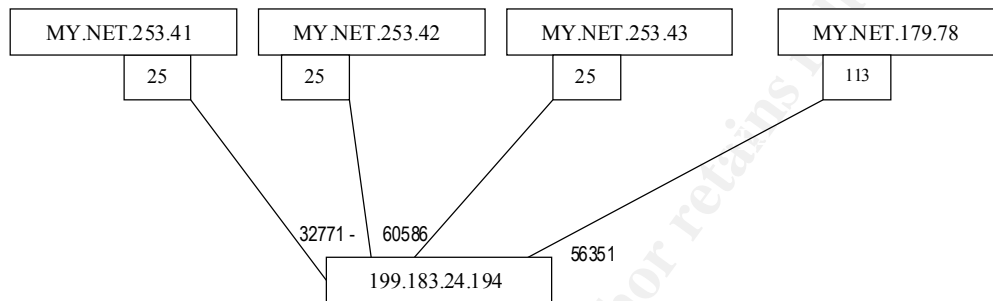


Figure 5 - OOS Link Analysis for top source

The dominant service is SMTP. In fact, nothing about this looks suspicious. What we see is a the typical high port to low port actions of a mail exchange, coupled with one IDENT request.

My guess here is that the packets have been considered OOS because the client machine (probably an upstream mail server) is running an ECN enabled Kernel. For this particular source, no other destination ports are used.

The excerpt of OOS log below demonstrates that the previously reserved bits are set, lending support to the ECN theory.

```

06/21-00:12:12.377948 199.183.24.194:33206 -> MY.NET.253.41:25
TCP TTL:54 TOS:0x0 ID:17668 DF
21S***** Seq: 0x4C3835FA Ack: 0x0 Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 258044158 0 EOL EOL EOL EOL
  
```

3.8 Anomalous Activity

3.8.1 Sub7

There is clear evidence that a significant number of hosts are infected with the Sub7 Trojan (based on inbound and outbound packets with matching port numbers)

```

06/24-08:35:27.441986 [**] Possible trojan server activity [**] 216.220.164.141:1904
-> MY.NET.217.62:27374
06/24-08:35:27.443873 [**] Possible trojan server activity [**] MY.NET.217.62:27374 -
> 216.220.164.141:1904
06/24-08:35:28.049709 [**] Possible trojan server activity [**] 216.220.164.141:1904
-> MY.NET.217.62:27374
06/24-08:35:28.050294 [**] Possible trojan server activity [**] MY.NET.217.62:27374 -
> 216.220.164.141:1904
06/24-08:35:28.650287 [**] Possible trojan server activity [**] 216.220.164.141:1904
-> MY.NET.217.62:27374
06/24-08:35:28.650590 [**] Possible trojan server activity [**] MY.NET.217.62:27374 -
> 216.220.164.141:1904
  
```

Without access to the rules that resulted in these alerts, it is not possible to say if every logged occurrence is genuine Sub 7 transaction rather than another protocol that happens to be using the ephemeral port 27374, but Sub7 does seem likely.

3.8.2 High port 65535 tcp - possible Red Worm

The Red Worm infects Linux based computers, utilising other well known techniques to achieve a compromise (lpd, rpc.statd etc).

The worm provides a shell bound to TCP port 65535 upon receipt of an appropriately crafted ICMP packet and attempts to deliver system password files to a number of email addresses.

Given this behaviour we can say with a reasonable degree of certainty that the host below is compromised and is attempting the mail delivery phase.

```
High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.253.24:65535 ->
199.154.149.191:25
High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.253.24:65535 ->
205.188.156.154:25
High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.253.24:65535 ->
209.196.123.3:25
```

Further details regarding this worm can be found at
<http://www.europe.f-secure.com/v-descs/adore.shtml>

3.8.3 Source / Destination Port 0

The source addresses below are all sending packets originating from TCP port 0.

24.31.91.225	192.168.1.1
24.101.139.227	195.202.182.128
24.166.172.31	202.131.99.142
24.226.209.39	213.118.91.234
24.252.172.9	

Also notice that at least one of the sources is using addressing defined in RFC1918.

3.8.4 Probable ECN TCP Flags

These sources are all sending packets with some combination of the "reserved" TCP flags set. This commonly indicates TCP/IP implementations using ECN.

```
24.113.101.87
66.24.29.12
217.0.71.215
```

Using a more recent version of Snort may prevent this alert from being reported, or help minimize the number of false positives, as it now (1.8+) understands ECN flags.

Further details regarding ECN can be found at <http://www.aciri.org/floyd/ecn.html>

3.8.5 Unknown TCP Options

The sources below are all sending packets with unknown TCP option values

24.101.54.192	202.131.99.142
62.59.8.127	211.220.73.227
64.228.216.149	212.242.234.14
62.243.160.209	213.51.179.80
111.111.111.111	217.0.72.73

3.8.6 SYN FIN Flags

The 64 source addresses below have each sent at least one packet with an illegal SYN FIN combination.

24.3.98.110	62.59.136.17	199.44.16.233
24.30.32.162	62.59.150.145	200.171.165.156
24.31.91.225	62.59.153.108	202.131.99.142
24.67.149.250	62.59.153.75	206.128.215.88
24.95.215.180	62.106.25.11	208.33.170.117
24.95.222.128	62.149.151.76	208.228.170.55
24.101.54.192	62.149.156.164	211.220.73.227
24.101.139.227	62.163.52.91	212.95.76.26
24.113.101.87	62.226.74.193	212.111.188.15
24.141.215.220	62.243.160.209	212.123.168.2
24.166.172.31	64.228.216.149	212.139.40.11
24.169.233.12	64.230.225.112	212.182.11.161
24.180.168.160	65.14.2.105	213.51.179.80
24.200.176.151	66.50.122.62	213.77.216.89
24.222.118.32	66.67.48.85	213.96.38.157
24.226.208.109	111.111.111.111	213.153.212.59
24.226.209.39	130.13.168.129	217.0.71.215
24.252.172.9	192.168.1.1	217.0.72.73
62.54.135.97	194.236.154.110	217.157.84.2
62.59.7.186	195.202.182.128	217.157.84.25
62.59.8.127	195.204.46.106	
62.59.16.29	196.21.162.250	

3.8.7 XMAS Packets

The sources below have sent packets with SFRPAU flag options set.

62.163.52.91	195.202.182.128
66.67.48.85	212.123.168.2
111.111.111.111	217.0.72.73
192.168.1.1	217.157.84.2

3.9 Defensive Recommendations

3.9.1 Ingress / Egress Filtering

The importance of ingress/egress filtering cannot be overstated. Much of the malicious activity detected would be stopped by aggressive access control lists on the Internet border routers.

Many of the basic, scripted attacks would be blocked, leaving both sensors and staff free to concentrate on the serious attempts and/or breaches. Ingress filters also serve a useful function in limiting the damage caused by directed broadcast attacks such as Smurf.

Given the frequency with which attacks are observed from the networks described in 3.3.2 and 3.3.7, it may be appropriate to block incoming traffic from these subnets at the border routers.

Packet filtering isn't the only answer, as the recent Code Red Worm has pointedly demonstrated. Systems must be kept up to date, and exploits are developed that make use of services that have to be let through a packet filter (after all there's little point in blocking TCP port 80 traffic to a web server) - as in the case of Code Red.

Any changes that impact the flow of traffic into and out of the site have to be researched thoroughly before implementation, but once a baseline has been decided upon, packet filters can remain reasonably static.

Don't forget that the goal to strive for is "drop everything, permit only what's needed".

3.9.2 Anti-Spoofing

Border routers should be configured in such a way as to reject traffic with spoofed addresses (though note the concerns in section 3.2 regarding multicast destination addresses)

As a simple policy, drop inbound packets where

- Source address is as defined by RFC1918 (private addresses such as 10.x.x.x)
- Source address is within MY.NET

Similarly, drop outbound packets where

- Source address is as defined by RFC1918
- Source address is **not** within MY.NET

This can be implemented as part of broader ingress/egress filtering, or as a standalone precaution.

Though spoofed traffic does not play a major part in the malicious traffic observed at MY.NET, the current configuration could allow the campus to be used as a major player in DDOS attacks.

3.9.3 Restrictive Firewall Policy

In some ways this is similar to 3.9.1, but here we are concerned with traffic that has already passed inspection by the packet filtering performed at the border routers.

The firewall installed at this point should be able to provide two major functions

- Some idea of state (that is, accept inbound packets only when a outbound request has been observed)
- Some level of protocol awareness. The firewall should be able to tell that the traffic on TCP port 80 is actually HTTP, and not something masquerading as HTTP.

Obviously logging is also a concern, and depending on your requirements a product supporting high-availability functions (either through hardware or software).

Once again, a policy of "drop everything unless specifically allowed" should apply. If you already run a restrictive firewall policy, then the majority of the exploits and Trojans that exist will be blocked here.

3.9.4 Email Attachment Scanning

Many of the alerts recorded by Snort relate to activity resulting from Trojan application. Unlike remote exploits that lead to a machine compromise, Trojans are typically executed (albeit unwittingly) by the end-user.

Frequently, Trojan applications are delivered as attachments to email messages, where ill-educated users execute them, unaware of the consequences.

Mail attachment scanning when implemented outside of the users control (IE at a mail server on the mail transit path) can help reduce the risk that these types of Trojan applications pose by stripping the applications from the message body. This form of scanning is also useful as a first line of defence against more traditional viruses.

3.9.5 Virus Scanning

While scanning for malicious attachments at an email gateway can help reduce the risk of introducing infected or Trojan applications into the network, rigorous and non-optional virus scanning at the desktop and server level is also important.

3.10 Analysis Process

Having read a number of previous submissions, I didn't consider using SnortSnarf to parse the alert files as I expected that the amount of data was too great to process even on a fairly decent machine.

Instead I wrote a number Perl programs to sort the different type of files (Portscans, Alerts and OOS).

The Portscan and OOS files are processed by simple utilities that sort by occurrence of source and destination address and then summarise. The one for the alerts has more functionality, including the ability to discard multicast traffic, include traffic breakdowns and more.

The results of the summaries produced were cross checked using command line equivalents (combinations of grep, cut, awk, sort, uniq and wc). Once interesting data had been discovered, further investigations were performed with ad-hoc Perl or command line expressions.

One other technique that I found useful was simply to cat a given file to the screen, and look for visual patterns as it scrolls by. Not terribly sophisticated I know, but you'd be surprised at what jumps out.

Internet search engines and security focused sites (Neohapsis, Technotronic, SecurityFocus, SecurityPortal and SANS) also play a large part in gaining insights into approaches and for particular correlations.

It's certainly worth downloading and reading many of the previous submissions as there are many worthwhile tips to be gained from people's prior experiences.

© SANS Institute 2000 - 2002, Author retains full rights.

4 Appendices

4.1 Apache Failed Document Script

```
#!/usr/bin/perl
#
# failed.pl
#
# Looks for server status codes in the 4xx and 5xx ranges,
# spits out a minimal amount of info and error counts.
#
# usage : failed.pl < access_log

# define field positions that we're interested in
$SRC_IP=0;
$URL=6;
$SERVER_RESULT_CODE=8;

# setup some error counters
$doc_errors=0;
$server_errors=0;

# read the log
while (<>) {
    # skip any comments
    next if /^#/;

    @fields=split;

    # look for 4xx error types
    if ($fields[$SERVER_RESULT_CODE] =~ /4\d\d/) {
        print "NOT_FOUND : $fields[$SRC_IP], $fields[$URL]\n";
        $doc_errors++;
    }

    # look for 5xx error types
    if ($fields[$SERVER_RESULT_CODE] =~ /5\d\d/) {
        print "SERVER_ER : $fields[$SRC_IP], $fields[$URL]\n";
        $server_errors++;
    }
}
print "#\n";
print "# $doc_errors 4xx errors, $server_errors 5xx errors\n";
print "#\n";
```

4.2 Non-Multicast Alert Breakdowns

4.2.1 Alert Summary

Top 10 Alert Summary				
%	Count	Alert Detail	Sources	Dests.
61.762	78438	UDP SRC and DST outside network	35	95
17.008	21600	Possible trojan server activity	2970	9149
14.131	17946	Watchlist 000220 IL-ISDNNET-99051	69	19
2.844	3612	External RPC call	15	1184
1.600	2032	SMB Name Wildcard	194	710
0.806	1023	connect to 515 from outside	8	638
0.452	574	Watchlist 000222 NET-NCFC	8	8
0.253	321	Queso fingerprint	23	32
0.233	296	Back Orifice	3	180
0.232	295	High port 65535 tcp - possible Re	15	18

99.32	127000	total alerts		
	38950	total portscans		

	165950	total events		

4.2.2 Top Sources

Top 10 Source Addresses		Alert breakdown for this host	
56240	63.250.213.124	56240	UDP SRC and DST outside network
18804	63.250.213.26	18804	UDP SRC and DST outside network
14370	212.179.79.2	14370	Watchlist 000220 IL-ISDNNET-990517
7050	129.170.104.19	7050	Possible trojan server activity
5606	216.220.164.141	5606	Possible trojan server activity
2893	212.179.47.70	2893	Watchlist 000220 IL-ISDNNET-990517
2113	63.250.213.147	2113	UDP SRC and DST outside network
1492	165.230.53.35	1492	SMB Name Wildcard
1176	211.152.241.1	1176	External RPC call
734	24.147.14.159	734	External RPC call
110478 events			

4.2.3 Top Destinations

Top 10 Destination Addresses		Alert breakdown for this host	
56240	233.28.65.62	56240	UDP SRC and DST outside network
18804	233.28.65.164	18804	UDP SRC and DST outside network
14379	MY.NET.218.198	14369	Watchlist 000220 IL-ISDNNET-990517
4675	216.220.164.141	10	WinGate 1080 Attempt
2894	MY.NET.97.175	4675	Possible trojan server activity
2738	129.170.104.19	2893	Watchlist 000220 IL-ISDNNET-990517
2113	233.40.70.17	1	Null scan!
583	233.28.65.222	2738	Possible trojan server activity
300	MY.NET.100.56	2113	UDP SRC and DST outside network
257	MY.NET.70.97	583	UDP SRC and DST outside network
		300	Watchlist 000222 NET-NCFC
		149	Watchlist 000220 IL-ISDNNET-990517
		64	Queso fingerprint
		38	Null scan!
		3	High port 65535 tcp - possible Red Worm
		2	Possible trojan server activity
		1	SYN-FIN scan!
102983 events			

Upcoming Training

Click Here to
{Get CERTIFIED!}



Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC503: Intrusion Detection In-Depth	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201709,	Sep 11, 2017 - Oct 18, 2017	vLive
Baltimore September 2017 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Scottsdale SEC503	Scottsdale, AZ	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Boston SEC503	Boston, MA	Oct 09, 2017 - Oct 14, 2017	Community SANS
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced