Global Information Assurance Certification Paper

GIAC LEVEL TWO CERTIFICATION PRACTICAL

Intrusion Detection Curriculum

John Topp
February 20, 2001

Assignment 1 – Network Detects

Detect 1 SYN-FIN Scan with IPID 39426

From Snort Scan file

10.04 00:49:07.541759 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.1.21:21
10.04 00:49:09.507029 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.1.119:21
10.04 00:49:10.460067 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.1.167:21
10.04 00:49:10.661729 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.1.177:21
10.04 00:49:10.946873 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.1.191:21
10.04 00:49:11.062005 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.1.197:21
10.04 00:49:12.282000 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.2.3:21
10.04 00:49:12.384606 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.2.8:21
10.04 00:49:12.902806 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.2.34:21
10.04 00:49:13.188915 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.2.48:21
.
. 1549 lines omitted
.
10.04 01:10:36.867831 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.253.219:21
10.04 01:10:36.948783 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.253.223:21
10.04 01:10:38.904548 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.66:21
10.04 01:10:39.226466 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.82:21
10.04 01:10:39.385926 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.90:21
10.04 01:10:40.020602 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.122:21
10.04 01:10:41.266266 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.184:21
10.04 01:10:41.624641 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.202:21
10.04 01:10:42.044015 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.223:21
10.04 01:10:42.224671 SYN-FIN_scan!_ 128.2.81.133:21 MY.NET.254.232:21

From Snort Alert File

"10.04 00:49:16.135987 128.2.81.133:21 MY.NET.1.21:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "
"**SF**** Seq: 0x511AE364 Ack: 0x4CF340E0 Win: 0x404"
"00 00 00 00 00 00 ......"
"=+ "
"10.04 00:49:18.022315 128.2.81.133:21 MY.NET.1.115:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "
"**SF**** Seq: 0x6D52ED0B Ack: 0x370021B3 Win: 0x404"
"00 00 00 00 00 00 ......"
"=+ "
"10.04 00:49:18.101432 128.2.81.133:21 MY.NET.1.119:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "
"**SF**** Seq: 0x6D52ED0B Ack: 0x370021B3 Win: 0x404"
"00 00 00 00 00 00 ......"
"=+ "
"10.04 00:49:18.937155 128.2.81.133:21 MY.NET.1.161:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "
"**SF**** Seq: 0x5AC91143 Ack: 0x2B3617F3 Win: 0x404"
"00 00 00 00 00 00 ......"
"=+ "
"10.04 00:49:19.054828 128.2.81.133:21 MY.NET.1.167:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "

"**SF**** Seq: 0x5AC91143 Ack: 0x2B3617F3 Win: 0x404"
"00 00 00 00 00 00 ......"
"=+ "
"10.04 00:49:19.255936 128.2.81.133:21 MY.NET.1.177:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "
"**SF**** Seq: 0x833B70D Ack: 0x1FB713FC Win: 0x404"
"00 00 00 00 00 00 ......"
"=+ "
"10.04 00:49:19.541655 128.2.81.133:21 MY.NET.1.191:21 "
"TCP TTL:34 TOS:0x0 ID:39426 "
"**SF**** Seq: 0x833B70D Ack: 0x1FB713FC Win: 0x404"
"00 00 00 00 00 00
.
.
.
2065 more events

Source of Trace:

SANS Dataset for the December 2000 Conference – It constituted the largest amount of data – I though an analysis would be in order.

Detect was generated by:

Snort Alert logs / Snort OOS logs

Probability the source address was spoofed:

Low – A SYN-FIN scan is a recognizance method to gather information / penetrating firewall / IDS evasion. Spoofing the source address would have no benefit.

Canonical name: 8TH-DWARF.REM.CMU.EDU

Trying 128.2.81 at ARIN
Carnegie-Mellon University (NET-CMU-NET)
5000 Forbes Avenue
Pittsburgh, PA 15213
US

Netname: CMU-NET
Netblock: 128.2.0.0 - 128.2.255.255

Description of Attack:

The attacker is scanning for the FTP service. There is strong evidence of crafted packets.

TTL=34 – (Ok, I'm stretching a bit here and making some assumptions) GIAC Enterprises is located in the Washington DC area. Doing a TraceRT back to the source (128.2.81.133), I count 13 hops consistently over several days and times. This would put the starting TTL at around 47, a value significantly different from known starting values for various operating systems. (MS=32, Linux=64 etc – More details on page 146 of Track 3: Intrusion Detection Immersion Curriculum 3.2)

ID=39426 – Non changing over the entire dataset. Under normal conditions, we would expect to see this field increment.

Flag Bits – An obvious sign, the Syn and Fin bits together do not occur naturally.

Sequence Number – The tool reuses sequence numbers. I see a loose correlation in that in many places, the sequence number is recycled 4 times consecutively and then discarded. The Snort OOS files show evidence of dropping packets (when compared to the Alert File) and makes this problematic to say with certainty. The sequence number has been observed to always change when jumping from one subnet to the other.

Attack mechanism:

The attacker is looking for Unix boxes running WU-FTPD to compromise. (Microsoft boxes would respond with a reset no matter if the port was opened or closed.).

There are any number of WUFTPD vulnerabilities. A search of http://cve.mitre.org for WUFTPD produced the following list;

CVE-1999-0075 PASV core dump in wu-ftpd daemon when attacker uses a QUOTE PASV command after specifying a username and password.
CVE-1999-0080 wu-ftp FTP server allows root access via "site exec" command.
CVE-1999-0081 wu-ftp allows files to be overwritten via the rnfr command.
CVE-1999-0368 Buffer overflows in wuarchive ftpd (wu-ftpd) and ProFTPD lead to remote root access, a.k.a. palmetto.
CVE-1999-0720 The pt_chown command in Linux allows local users to modify TTY terminal devices that belong to other users.
CVE-1999-0878 Buffer overflow in WU-FTPD and related FTP servers allows remote attackers to gain root privileges via MAPPING_CHDIR.
CVE-1999-0879 Buffer overflow in WU-FTPD and related FTP servers allows remote attackers to gain root privileges via macro variables in a message file.
CVE-1999-0880 Denial of service in WU-FTPD via the SITE NEWER command, which does not free memory properly.
CVE-1999-0955 Race condition in wu-ftpd and BSDI ftpd allows remote attackers gain root access via the SITE EXEC command.
CVE-1999-0997 wu-ftp with FTP conversion enabled allows an attacker to execute commands via a malformed file name that is interpreted as an argument to the program that does the conversion, e.g. tar or uncompress.
CAN-1999-0076 ** CANDIDATE (under review) ** Buffer overflow in wu-ftp from PASV command causes a core dump.
CAN-1999-0156 ** CANDIDATE (under review) ** wu-ftpd FTP daemon allows any user and password combination.
CAN-1999-0661 ** CANDIDATE (under review) ** A system is running a version of software that was replaced with a Trojan Horse at its distribution point, e.g. TCP Wrappers, wuftpd, etc.
CAN-1999-0911 ** CANDIDATE (under review) ** Buffer overflow in ProFTPD, wu-ftpd, and beroftpd allows remote attackers to gain root access via a series of MKD and CWD commands that create nested directories.
CAN-2000-0573 ** CANDIDATE (under review) ** The reply function in wu-ftpd 2.6.0 and earlier does not properly cleanse an untrusted format string, which allows remote attackers to execute arbitrary commands via the SITE EXEC command.
CAN-2001-0138 ** CANDIDATE (under review) ** private pw program in wu-ftpd before 2.6.1-6 allows local users to overwrite arbitrary files via a symlink attack.

Correlations:

I didn't realize it until I started, but the ID field seems to be THE key to correlating. I did a search of the Sans web site and came up with 39426 matches. The following represents detects analyzed during the same time frame as the snort data used in this exercise (October and November 2000).

http://www.sans.org/y2k/102800.htm
http://www.sans.org/y2k/110100-1230.htm

In fact, further analysis of the snort data indicate that this signature is very popular.

It is interesting to note that the tool in use appears to have the ability to be modified as to what destination port to look at, but keeps the same behavior of source port = to destination port. Others have noticed port 9704 in play. (I analyze the significance of 9704 later)

Evidence of Active targeting:

I need to say yes. The attacker is not sweeping the network IP by IP. He or she seems to be targeting specific hosts. I searched the data every which way to find signs of a correlated attack or some discernable interleaving but could not find any. I'm forced to the conclusion that the attacker has pre-existent knowledge of who he was going after even though there is no evidence that every node he or she targeted was running FTP services. Maybe a previous scan (outside of this dataset) identified UNIX flavored machines and this is a honing in process.

Severity:

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

Criticality – 3 - FTP is not a critical network service but I do place it above a simple desktop system compromise.

Lethality – 5 - The listed exploits can grant Root

System –3- Tough to call since I have no real data on how the desktops are configured. Considering that this network seems to have lack security to begin with – a three feels right.

Net Countermeasures – 1 – There is no indication of any firewall

(3 + 5) – (3 + 1) = 4 - <- And I believe I am being generous.

Defensive Recommendations:

Evaluate the need for FTP. If not needed, disable the services. Ensure any needed FTP is well patched to current revision levels.

Multiple choice test question:

What is the default TTL for Windows NT SP6?

A. 64
B. 128
C. 255
D. 60

Answer = B
(Yea I know, it's different from the book but this is observed behavior on my machines)

Detect 2 – Possible Firewall Footprint

(No IPs were hurt in the sanitation of this trace)

Dec 31 02:49:08.319 Firewall-b kernel: 120 ICMP Info: Not sending ICMP Unreachable in response to non-information ICMP (210.57.16.44->MY.NET.5.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.5.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}) received on interface MY.FW.1.1

Dec 31 04:09:40.562 Firewall-b kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.FW.1.1)

Dec 31 06:58:43.060 Firewall-b kernel: 120 ICMP Info: Not sending ICMP Unreachable in response to non-information ICMP (210.57.16.44->MY.NET.5.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.5.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}) received on interface MY.FW.1.1

Dec 31 07:11:16.234 Firewall-b kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.FW.1.1)

Dec 31 08:17:02.182 Firewall-b kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.FW.1.1)

Dec 31 09:09:03.371 Firewall-b kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.FW.1.1)

Dec 31 12:21:54.445 Firewall-b kernel: 120 ICMP Info: Not sending ICMP Unreachable in response to non-information ICMP (210.57.16.44->MY.NET.5.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.5.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}) received on interface MY.FW.1.1

Dec 31 05:27:40.732 Firewall kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.NET.4.1)

Dec 31 09:34:04.487 Firewall kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.NET.4.1) [29 kernel log messages at level 2 suppressed]

Dec 31 14:40:52.746 Firewall kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.NET.4.1)

Dec 31 16:30:18.001 Firewall kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.NET.4.1)

Dec 31 20:46:08.251 Firewall kernel: 226 IP packet dropped (210.57.16.44->MY.NET.2.0: Protocol=ICMP[Time exceeded (in transit)] {Inner: MY.NET.2.0->ci582208-a.ganvil1.ga.home.com[24.12.76.66]: Protocol=ICMP[Echo request]}): dest is broadcast address (received on interface MY.NET.4.1)

Source of Trace:

My network at my place of employment.

Detect was generated by:

Raptor Firewall

Probability the source was spoofed:

No lookup for the 210.57.16.44 address, however the block is owned by;

```
whois -h whois.apnic.net 210.57.16.44 ...

% Rights restricted by copyright. See http://www.apnic.net/db/dbcopyright.html

inetnum:    210.57.16.0 - 210.57.16.255
netname:    BELLNET00001
descr:      The company is involved in supplying internet services.
descr:      They mainly deal with internet connectivity to
descr:      personal user in Japan.
descr:      Their contents or service include
descr:      internet auction, membership Web Sites
descr:      (Main contents is related an entertainer),
descr:      supply company mailing service, and
descr:      internet direct mailing service.
country:    HK
admin-c:    TL108-AP
tech-c:     TL108-AP
mnt-by:     MAINT-AP-LEVEL3
changed:    cusfacprov.hk@level3.com 20000904
source:     APNIC
```

The other outside address –

```
nslookup 24.12.76.66
Canonical name: ci582208-a.ganvil1.ga.home.com
```

I do believe the 210 address is spoofed. The 24 address looks to be where the hostile is really sitting.

Description of attack:

It looks to be any number of things, but I'm going to settle on an attempt to footprint the Firewall.

Attack mechanism:

The attack failed because ultimately our Firewall doesn't respond with ICMP. However, let's consider what the packets would do against a Firewall that does allow ICMP (outbound) error messages. (Assume that the firewall does not allow ping – informational - replies)

The packets have two different settings that will elicit a number of different replies; Broadcast destination and TTL=0. The packets are sent to both a network that does exist and a network that does not exist. Since only 1 type of ICMP error message will be sent back for each network, such information can be used to deduce the order in which the stack processes ICMP conditions. (i.e. Time Exceeded before or after Network Unreachable) The reply – and possibly lack of reply, may give the attacker a 'footprint' as to how the stack works and therefore allows for the possibility of creating a database that compares the different firewall vendors. We see some evidence of this possibility in the above trace. For networks that exist, we see the stack keying off of the "IP packet dropped dest is broadcast address". If the network does not exist, we see the stack keying off of "Not sending ICMP Unreachable in response to non-information ICMP"

The different Source and Reply addresses (210 and 24) had me stumped until I applied some source routing logic to it. If the attacker spoofed the source IP (210) and inserted source routing into the original ping packet, the reply would be sent via the specified source route. Page 105 of TCP/IP Illustrated Volume 1 adds credence to this.

> *"The Host Requirements RFC specifies that a TCP client must be able to specify a source route, and that a TCP server must be able to receive a source route, and use the reverse route for all segments on that TCP connection."*

The time frame of the attack also wasn't lost on me. December 31st ! Perhaps the extra added effort on spoofing was designed to catch the analyst of guard.

Correlations:

I searched several different sites but could not come up with any. To be sure, I found loads of information on OS fingerprinting and Firewalking but this seems to be more of a 'application' fingerprinting. I assume I'm not looking in the right places. Any feedback would be much appreciated.

Evidence of active targeting:

I would think yes, the setup is very methodical.

Severity:

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

Criticality – 5 - Firewall

Lethality – 5 - Indeed, given the right exploit, root can be obtained

System –4- Tough to call since I have no real data on how the firewall is configured. My request for such data was denied.

Net Countermeasures – 5 – Very restrictive

(5 + 5) – (4 + 5) = 1

Defensive Recommendation:

As mentioned, our Firewall does not allow informational or error ICMP replies. I would advise a close monitoring of security listservers and Bugtraq to be on top of any new exploits.

Multiple choice test question:

Which of the following is NOT a ICMP type 3 code

A. Port Unreachable
B. Communication Administratively Prohibited
C. Echo Reply
D. Fragmentation needed but DF bit set

Answer C

Detect 3 – VBS Virus Encounter

09/23/00 22:02:22.000
IP: ID = 0xD561; Proto = TCP; Len: 48
IP: Time to Live = 125 (0x7D)
IP: Source Address = 12.78.117.145
IP: Destination Address = 12.78.120.247
TCP: ....S., len: 8, seq: 9715577-9715584, ack: 0, win: 8192, src: 2831 dst: 135

09/23/00 22:02:24.157
IP: ID = 0xDA61; Proto = TCP; Len: 48
IP: Time to Live = 125 (0x7D)
IP: Source Address = 12.78.117.145
IP: Destination Address = 12.78.120.247
TCP: ....S., len: 8, seq: 9715577-9715584, ack: 0, win: 8192, src: 2831 dst: 135

09/23/00 22:02:30.147
IP: ID = 0xDD61; Proto = TCP; Len: 48
IP: Time to Live = 125 (0x7D)
IP: Source Address = 12.78.117.145
IP: Destination Address = 12.78.120.247
TCP: ....S., len: 8, seq: 9715577-9715584, ack: 0, win: 8192, src: 2831 dst: 135

09/23/00 22:02:42.153
IP: ID = 0xED61; Proto = TCP; Len: 48
IP: Time to Live = 125 (0x7D)
IP: Source Address = 12.78.117.145
IP: Destination Address = 12.78.120.247
TCP: ....S., len: 8, seq: 9715577-9715584, ack: 0, win: 8192, src: 2831 dst: 135

Others just like it from different source IP's but on the same 12.78.117 network

September 25, 2000	22:02
September 25, 2000	23:09 & 23:29
October 03, 2000	16:10
October 21, 2000	01:56
October 23, 2000	14:36 & 14:58

Source of Trace:

My Home Internet Computer

Detect was generated by:

BlackICE – data was then read into Microsoft Network Monitor

Probability the source address was spoofed:

There is no need to spoof.

Canonical name: 145.arlington-28-29rs.va.dial-access.att.net
NetBIOS Name: MOBILE1 (consistently observed during attack time)

Description of attack:

Hostile was attempting to ascertain if my Windows 98 box has File and Printer sharing bound to my dial up interface.

Attack Mechanism:

Port 135 is host the Microsoft NetBIOS Service (MS Exchange RPC End-Point Mapper). One would query this service to learn of any shared directories being offered by the target. Although the default for Windows 9X boxes is to turn this off on the Dial connection, many users mis-configure this and wind up sharing directories or entire drives out to the Internet. A scan of my subnet (for learning purposes only) indicates that Windows NT users are even more prone to be sharing. Unfortunately, NT right out of the box, sets no real security on the file system and user rights allow the EVERYONE group full access. Entry is a mouse click away (with Legion) – scary.

In this case, the hostile is probably a victim themselves. Last year we saw the rise of VBS viruses that wisk their way into a system via Microsoft Office applications. Once a user invokes the document, the macro virus runs and infects the host system and variety of ways. Some Viruses in this class do not even need Email to spread, they infect by hopping from share to share. Here are some methods in which these virus propagate:

W32 / QAZ worm
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\StartIE=
C:\WINDOWS\notepad.exe qazwsx.hsq
When ever the user runs NOTEPAD, the worm is executed and this then runs NOTE.COM.

VBS / NetLog.worm.c, W95/Firkin.worm, et all
Copies a VB script to the Startup folder – when a user logs on or reboots, the script is run.

W32 / Msinit
Modifies the Win.ini load = lines to load itself on the next boot up where the installation finishes by installing itself as a service with the registry key -
HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices\msinit

No matter the method of infection, all these viruses will then scan reachable subnets looking for 'open' shares. Once found, the virus will install itself (replicate) and continue on it's merry way.

Depending on the author, these worms have a wide range of actions. The QAZ worm listens on Port 7597 (TCP) and may be acting as a backdoor to load more advanced Trojans such as Back Orifice, SubSeven, or simply, a password stealer. W97M/Thus.CH will corrupt important Operating System Files affecting operations. W95/Firkin.worm would either command your modem to dial 911 or format your hard drives. VBS and Macro scripts are very versatile and can be coded to do most anything.

Correlations:

Much has been written about this class of virus. The following web sites offer more in-depth analysis:
WWW.MCAFEE.COM
http://www.symantec.com/avcenter/

If I had to make a guess, I would say that MOBILE1 was infected with either the MSINT or QAZ worms. I base my presumption solely on the time frame. As evident in the traces, the virus was active on MOBILE1's machine in the September to October time frame. MCAFEE posted the alert for QAZ on August 8th and MSINT on September 28th.

In this case, MOBILE1 had open C$, ADMIN$ and D$ shares (Windows NT default Admin shares) – neither were protected by passwords or NTFS, the volume was FAT. <shaking head> Both Viruses would have had ample opportunity to infect.

Evidence of active targeting:

None, the method of attack and infection is opportunistic and takes place without the owners knowledge.

Severity:

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

Criticality – 5 - It was my system and the gateway into my home net – and I'm not that timely on doing backups.

Lethality – 5 - Indeed, given the right exploit, root can be obtained.

System – 4 - Doesn't seem right to give it a five. Although I have set my antiviral protection to update daily, the window of between new viruses and new definition files always creates a hole. Besides, I have children, <grumble> they lack a certain caution about what they open and do on the Internet.

Net Countermeasures – 5 – Firewall, No file and print sharing bound to the dial up – it's tight.

(5 + 5) – (4 + 5) = 1 – Sounds about right, considering how hard it is to keep up with the new threats.

Defensive Recommendation:

Boot the kids out of the house. <grin>
None for me but for MOBILE1 the following;

1. Rebuild your system – you may be "owned" several times over by now.
2. Disable bindings for WINS client on external interface.
3. Remove the Everyone group from the Remote login right.
4. Disable anonymous logins
   HKLM\System\CurrentControlSet\Control\LSA\RestrictAnonymous=1
5. Install Antiviral protection.
6. Use caution when opening any email attachments.
7. Install a personal firewall – BlackICE or Zone Alarm

The key is to lock down the system, these are just the basics.

Multiple choice test question:

Which Ports would you block to prevent NeBIOS activity?

A. 135
B. 137
C. 138
D. 139
E. All of the above

Answer E

Detect 4 WEBOT encounter

(No IPs were hurt in the sanitation of this trace)

Dec 7 18:51:20.051 Firewall-b httpd[9245]: 121 Statistics: duration=0.27 id=iDjQG sent=336 rcvd=604 srcif=hme0 src=164.124.250.222/59650 dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 19:29:34.585 Firewall-b httpd[9245]: 121 Statistics: duration=0.06 id=iDpMR sent=107 rcvd=604 srcif=hme0 src=209.73.164.13/57216 srcname=vscooter.sv.av.com dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 20:46:45.529 Firewall-b httpd[9245]: 121 Statistics: duration=3.55 id=iDy7d sent=262 rcvd=604 srcif=hme0 src=202.84.172.230/41632 dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 20:55:53.329 Firewall-b httpd[9245]: 121 Statistics: duration=0.18 id=iDyHX sent=172 rcvd=604 srcif=hme0 src=213.216.143.39/31934 dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://www.MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:11:33.510 Firewall-b httpd[9245]: 121 Statistics: duration=3.29 id=iDzee sent=198 rcvd=623 srcif=hme0 src=206.160.169.41/1785 dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:12:55.926 Firewall-b httpd[9245]: 121 Statistics: duration=0.57 id=iDzfR sent=208 rcvd=623 srcif=hme0 src=212.246.31.244/64325 srcname=www.dokumentori.fi dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:13:43.555 Firewall-b httpd[9245]: 121 Statistics: duration=5.16 id=iDzhB sent=133 rcvd=604 srcif=hme0 src=204.123.28.31/4419 srcname=atrax1.pa-x.dec.com dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://www.MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:27:39.101 Firewall-b httpd[9245]: 121 Statistics: duration=0.07 id=iDAc1 sent=120 rcvd=604 srcif=hme0 src=213.41.126.253/1666 srcname=micropole-253.126.rev.fr.colt.net dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:27:39.862 Firewall-b httpd[9245]: 121 Statistics: duration=0.00 id=iDAc3 sent=120 rcvd=604 srcif=hme0 src=213.41.126.253/1672 srcname=micropole-253.126.rev.fr.colt.net dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:43:40.364 Firewall-b httpd[9245]: 121 Statistics: duration=0.10 id=iDAsZ sent=217 rcvd=623 srcif=hme0 src=208.41.25.199/1477 dstif=qfe0 dst=MY.NET.2.210/80 dstname=www.MY.WEB.COM op=GET arg=http://MY.WEB.COM/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:46:38.679 Firewall-b httpd[9245]: 121 Statistics: duration=0.04 id=iDAve sent=254 rcvd=623 srcif=hme0 src=208.41.25.199/1891 dstif=qfe0 dst=MY.NET.2.4/80 dstname=www.MY.WEB2.COM op=GET arg=http://aiep.state.gov/robots.txt result="404 Object Not Found" proto=http rule=103

Dec 7 21:47:29.056 Firewall-b httpd[9245]: 121 Statistics: duration=0.01 id=iDAya sent=220 rcvd=623 srcif=hme0 src=208.41.25.199/1982 dstif=qfe0 dst=MY.NET.2.209/80 dstname=www.MY.WEB3.COM op=GET arg=http://library.state.gov/robots.txt result="404 Object Not Found" proto=http rule=103

Source of Trace:

My network at my place of employment.

Detect was generated by:

Raptor Firewall

Probability the source was spoofed:

For reasons covered later, no, these sources were not spoofed. Owners of source addresses as follows:

164.124.250.222 – no lookup, block owned by;
Trying 164.124.250 at ARIN
DACOM Corporation (NET-DACOM-BORANET)
706-1, Yeoksam-dong,Kangnam-Ku
Seoul, 135-610
KR

209.73.164.13 - Canonical name: vscooter.sv.av.com
Block Owned by;
Trying 209.73.164 at ARIN
AltaVista Company (NETBLK-INTERNET-BLK-1-AV)
529 Bryant St.
Palo Alto, CA 94301
US

202.84.172.230 - Canonical name: mailgate.cwhkt.com
Block Owned by;
Trying 202.84.172 at ARIN
Redirecting to APNIC ...
Trying 202.84.172 at APNIC
Trying 202.84 at APNIC
inetnum: 202.84.0.0 - 202.84.7.255
netname: CIC-HK
descr: China Internet Corporation
descr: 1/F Xinhua News Agency Building
descr: 5 Sharp Street West, Wanchai
country: HK
admin-c: AM57-AP
tech-c: AM57-AP
mnt-by: APNIC-HM
mnt-lower: MAINT-CIC-AP
changed: hostmaster@apnic.net 20000519
source: APNIC

213.216.143.39 – no lookup
Block owned by
Trying 213.216.143 at ARIN
Redirecting to RIPE ...
Trying 213.216.143 at RIPE

inetnum: 213.216.143.0 - 213.216.144.255
netname: INKTOMI-UK-NET
descr: Inktomi Corp UK
country: GB
admin-c: JM165-RIPE
tech-c: CLAU1-RIPE
status: ASSIGNED PA
notify: clau@exodus.net
mnt-by: EXODUS-MNT
changed: clau@exodus.net 20001101
source: RIPE

206.160.169.41 – no lookup
Block owned by
Trying 206.160.169 at ARIN
SprintLink (NETBLK-SPRINT-S)
VAHRNA0402
13221 Woodland Park Road
Herndon VA 22071
Netname: SPRINT-S
Netblock: 206.160.0.0 - 206.160.255.255
Maintainer: SPRN

212.246.31.244 - Canonical name: www.dokumentori.fi
Block owned by
Trying 212.246.31 at ARIN
Redirecting to RIPE ...
Trying 212.246.31 at RIPE
inetnum: 212.246.31.0 - 212.246.31.127
netname: SME-TAMPEREENTYOVAENTEATTERIOY
descr: TAMPEREEN TYOVAEN TEATTERI OY
descr: TAMPERE, FINLAND
country: FI
admin-c: MU512-RIPE
tech-c: TV227-RIPE
status: ASSIGNED PA
notify: registry@tpo.fi
changed: registry@tpo.fi 19990813
source: RIPE

204.123.28.31 - Canonical name: atrax1.pa-x.dec.com
Block Owned by
Trying 204.123.28 at ARIN
Digital Equipment Corporation (NETBLK-DEC-P)
Digital Equipment Corporation
Network Systems Laboratory
250 University Avenue
Palo Alto, CA 94301-1616
Netname: DEC-P
Netblock: 204.123.0.0 - 204.123.255.255

213.41.126.253 - Canonical name: micropole-253.126.rev.fr.colt.net
Block Owned by
Trying 213.41.126 at ARIN

Redirecting to RIPE ...
Trying 213.41.126 at RIPE
inetnum: 213.41.126.0 - 213.41.126.255
netname: FR-MICROPOLE
descr: FR-MICROPOLE
country: FR
admin-c: LJ498-RIPE
tech-c: TT997-RIPE
status: ASSIGNED PA
mnt-by: COLT-FR-MNT
mnt-lower: COLT-FR-MNT
changed: tarik.tifour@fr.colt.net 20001116
source: RIPE

208.41.25.199 – no lookup
Block owned by
whois -h whois.arin.net !netblk-ien-estand ...
E-Standard, Inc. (NETBLK-IEN-ESTAND)
4221 Redwood Ave.
Marina Del Rey, CA 90066
US
Netname: IEN-ESTAND
Netblock: 208.41.25.192 - 208.41.25.223

Description of attack:

This is not an attack! Call me a neophyte but I knew nothing about robots.txt. When I started searching for information on it, it was difficult to come by (0 hits at both SANS and CERT web sites). As such, I decided to include it as one of the detects in order to share information.

The full text for what ROBOT.TXT is and what it is used for can be found at:
http://info.webcrawler.com/mak/projects/robots/norobots.html

To paraphrase –

Some search engines, such as AltaVista (209.73.164.13 - Canonical name: vscooter.sv.av.com) use 'robots', to index Internet sites for their search databases. Robots.txt, is a file that in theory, contains a list of directories and documents on the web server that should not be included in the 'bots' search. It should work as an exclusion file.

It is important to note, that this is not a "standard". It is a gentlemen's agreement on the way things should work. By no means do all 'bots' honor the system.

In of itself, there is no outright danger with allowing a bot into your site to do indexing, in fact, if your running a public web server, it is beneficial. However, poorly secured web servers can very well find their CGI-Scripts directory full indexed and included in dozens of databases. This would not be a good thing.

Attack mechanism:

No big deal here. Any user can issue a GET for http://{Some Web Server}/robots.txt and receive the file if it exists on the Web Server. Web Crawlers, or Spiders as they are sometimes known, automate this process. The spider then proceeds to load all pages that are not denied by robots.txt looking for meta-tags to use in it's index.

Correlations:

A list is maintained at:
http://info.webcrawler.com/mak/projects/robots/active.html
that currently contains 254 entries in a Web Robots Database. By no means is it considered to be the last word in webots, but it's a start.

A Webot FAQ can be found at;
http://info.webcrawler.com/mak/projects/robots/faq.html

Evidence of activity targeting:

No – Opportunistic process.

Severity:

This was not an attack. Any problems directly result from improperly configured web servers.

Defensive recommendation:

None, since Webbots are not looking for robots.txt they would appear as ordinary traffic, it would be a waste of time blocking those that do.

Multiple choice test question:

Where would you find the file Robots.txt on a web server?

A. /cgi-bin
B. /
C. /Winnt
D. /public

Answer B

Assignment 2 – Analyze This

This data analysis covers the period from September 27, 2000 through November 23, 2000. Unfortunately, due to intermittent problems on your network, the data is not contiguous over this period. Never the less, enough data was collected to form a solid analysis of the state of your network.

Three different criteria, Scans, Alerts, and OOS (Out Of Spec) were used in the data capture by Snort. The Scan files indicate behavior by a source directed at a target indicative of 'foot printing' – the method an attacker will use to identify what nodes (computers) are operating on your network. Depending on which ports are targeted, scans may identify which platform (Microsoft or UNIX) are in operation.

Alert files indicate sources that are sending more specific probes into your network. Based on signatures, they will indicate if someone is trying to identify what version OS is running, ports they are interested in, scans for well known Trojans, etc. We will rely on these heavily to ascertain if any compromises have taken place.

OOS files reveal packets that are mangled in some way. They are mostly used for correlation when one suspects an attempted or full compromise of a target system since they capture the payload data.

I'm going to use a color convention as we proceed through the data analysis. Blocks of yellow are warnings that the IPs contained within need to be examined due to suspicious activity. Blocks of red indicate a highly probable compromise has taken place or will take place.

SCAN FILE STATISTICS – SOURCE IP

Snort identified 1231 unique source addresses generating 310,477 packets. The following table enumerates the top talkers. This table also represents the top talkers who are responsible for 27.4% of all Scan packets captured (84,981 packets).

Source IP	Events	Lookup
66.9.27.254	20649	No reverse DNS (US)
62.252.21.241	13057	Canonical name: pc241-gui4.cable.ntl.com (Great Britain)
194.244.78.145	11904	No reverse DNS (Italy)
63.88.175.201	11718	Canonical name: www.multilateral.com (US)
62.157.23.237	9641	Canonical name: p3E9D17ED.dip.t-dialin.net (Germany)
63.249.55.245	9073	No reverse DNS (Germany)
62.96.169.86	8939	Canonical name: m-dialin-86.addcom.de (Germany)

Top SCAN Source IP addresses

66.9.27.254 Performed a massive SYN scan on November 23 between 19:39 and 19:42. The scan spanned 120 different class C address spaces and all packets were targeted to port 515.

WWW.SANS.ORG, an awesome think-tank of some of the best and brightest in the universe, issued an alert on 515 scans on November 20th.

Ref: http://www.sans.org/newlook/alerts/port515.htm

To paraphrase these guardians of network security - Port 515 host the Unix LPR service, which developed vulnerabilities in the scan time frame. The attacker was fishing for a box to compromise.

62.252.21.241 Performed a massive SYN scan on October 08 between 08:03 and 09:01. The scan spanned 89 different class C address spaces and with the exception of 3 packets all other packets were targeted to port 21. Needless to say, this hostile was looking for the FTP service to compromise. The three rouge packets look to be the result of noise.

194.244.78.145 Performed a SYN scan (Port) on November 04 between 03:53 and 04:02. The scan was directed at a single target, MY.NET.220.2 and checked every port from 1 through 2583. This makes me a bit concerned. The hostile appears to have no interest in any other machine on your network – a sign that the recognizance work has already been done. I advise that this host be checked over – to at least ascertain what is so interesting about this target to this hostile.

MY.NET.220.2

63.88.175.201 Performed a massive SYN scan on October 29 between 08:56 and 10:03. The scan spanned 118 different class C address spaces and with the exception of 1 packet, all packets were targeted to port 21. Again, another FTP scan but it appears to have hit pay-dirt Under certain conditions a Microsoft DNS server can be configured to query the Name Service Port on a distant node.

> Quote from TechNet - PSS ID Number: Q173161
>
> There is also a WINS-R or WINS Reverse Lookup entry that can be added to the reverse zone. Because WINS does not have a reverse lookup capability, however, this record instructs the DNS server to perform a NetBIOS node adapter status lookup, or an NS Query, against the host. (Note – the destination ports are all above 1024 indicating this is a SAMBA box, Microsoft would use port 137 for source and destination)

Such traffic patterns may also result from a DNS server that is not responding in due time.

Ref: http://www.robertgraham.com/pubs/firewall-seen.html#10.6

There is a chance that the traffic is normal, however the surround circumstances are not (Port 21 scan). As such, I will classify this as hostile. Check this machine and the next for signs of compromise.

10.29	09:51:24	63.88.175.201	3669	MY.NET.115.97	21	SYN	**S*****
10.29	09:51:24	63.88.175.201	3670	MY.NET.115.98	21	SYN	**S*****
10.29	09:51:24	63.88.175.201	137	MY.NET.115.150	1589	UDP	
10.29	09:51:25	63.88.175.201	3711	MY.NET.115.139	21	SYN	**S*****
10.29	09:51:25	63.88.175.201	3707	MY.NET.115.135	21	SYN	**S*****

62.157.23.237 Performed a massive SYN scan on October 28 between 06:58 and 07:31. The scan spanned 118 different class C address spaces and with the exception of 2 packets all were targeted to port 21. Again, we see port 137 UDP packets. You may also notice that MY.NET.115.150 is involved again.

10.28	07:16:36	62.157.23.237	137	MY.NET.115.150	4079	UDP
10.28	07:13:45	62.157.23.237	137	MY.NET.98.86	4796	UDP

63.249.55.245 Performed a series of focused UDP scans of 11 nodes on your network.

September 28th	UDP Scan MY.NET.209.242	ports 1052 – 1124
October 1st	UDP Scan MY.NET.202.10	ports 1076 & 1077
	UDP Scan MY.NET.206.106	ports 1182 & 1183
	UDP Scan MY.NET.211.254	ports 4 different
October 11th	UDP Scan MY.NET.205.214	ports 2707 – 3003
October 21st	UDP Scan MY.NET.206.94	ports 2000 – 2359
October 30th	UDP Scan MY.NET.204.18	ports 8 different
	UDP Scan MY.NET.205.246	ports 7 different
	UDP Scan MY.NET.212.114	ports 7 different
	UDP Scan MY.NET.215.210	ports 12 different

All of the source ports are either 7777 or 7778. A search of the Internet indicates that the game Unreal utilizes these two UDP ports. Scanning the ports over the course of a few days with NMAP indicates a constant presence; the traffic appears to be nothing more than users abusing your systems and Internet connection.

Ref: http://www.robertgraham.com/pubs/firewall-seen.html

62.96.169.86 Performed a massive SYN scan on October 25th between 08:20 and 09:22. The scan spanned 117 different class C address spaces with all packets directed to port 21. Looks to be a plain vanilla FTP scan.

SCAN FILE STATISTICS – DESTINATION IP

Snort identified 35,844 unique destination addresses receiving 310,454 packets. The following table enumerates the top targets. This table also represents the top targets that are responsible for 8.6% of all Scan packets captured (26,892 packets).

Top Targets	Events
sMY.NET.220.2.txt	11926
sMY.NET.218.50.txt	2359
sMY.NET.253.114.txt	1976
sMY.NET.206.94.txt	1799
sMY.NET.162.77.txt	1759
sMY.NET.120.36.txt	1591
sMY.NET.205.214.txt	1589
sMY.NET.215.210.txt	1367
sMY.NET.60.16.txt	1306
sMY.NET.140.57.txt	1220

Top Scan Targets

ALERT FILE STATISTICS – SOURCE IP

Snort identified 2488 unique source addresses generating 116,770 packets. The following table enumerates the top talkers. This table also represents the top talkers who are responsible for 26.7% of all Alert packets captured (31,287 packets).

Source IP	Events	Lookup
160.78.49.191	7224	Canonical name: ema.chim.unipr.it (Italy)
208.61.4.207	6659	Canonical name: adsl-61-4-207.mia.bellsouth.net (US)
159.226.45.3	6295	Canonical name: aphy.iphy.ac.cn (China)
212.179.95.5	6117	Canonical name: cable-95003.bezeqint.net (Israel)
209.92.40.32	4992	Canonical name: dslcv1-32.fast.net (US)

160.78.49.191 – Shows a propensity for Port Scans and SYN+FIN scans. All scans originate from port 53 and are targeted to port 53. Single day visitor on September 30th 13:10 to 13:52. Attacker is looking for DNS servers to compromise.

208.61.4.207 - Shows a propensity for Port Scans and SYN+FIN scans. All scans originate from port 9704 and are targeted to port 9704. Single visit under this IP – October 2nd 06:38 to 06:51. I found some interesting data on port 9704.

Ref: http://lists.insecure.org/incidents/2000/Sep/0054.html
Ref: http://lists.insecure.org/incidents/2000/Sep/0057.html

The data suggests that such a scan (port 9704) indicates a hostile source looking for a target already compromised with a wu-ftpd exploit or inetd exploit.

159.226.45.3 – Alerts generated as a result of WATCHLIST 000222 NET NCFC. Source shows a narrow interest in Port 25. This can be normal – if your mail servers need to do business with Institute of Computing Technology Chinese Academy of Sciences in Beijing, but considering the source, I think not. The following traces indicate probable compromises.

09.26	02:10:53.639433	159.226.45.3	3132	MY.NET.6.7	25
09.27	02:11:50.845918	159.226.45.3	2402	MY.NET.6.7	25
09.27	02:11:50.942189	159.226.45.3	2402	MY.NET.6.7	25
09.27	04:39:44.448634	159.226.45.3	113	MY.NET.6.7	1245
09.27	04:39:46.092083	159.226.45.3	2800	MY.NET.6.7	25
09.27	04:39:50.641620	159.226.45.3	2800	MY.NET.6.7	25
09.27	04:40:07.429855	159.226.45.3	2800	MY.NET.6.7	25
10.04	09:49:47.278962	159.226.45.3	4082	MY.NET.6.7	25
10.04	09:49:47.347407	159.226.45.3	4082	MY.NET.6.7	25
10.04	09:49:47.427893	159.226.45.3	4082	MY.NET.6.7	25
10.04	09:53:14.426035	159.226.45.3	113	MY.NET.6.7	21555
10.04	09:53:15.849293	159.226.45.3	4090	MY.NET.6.7	25
10.04	09:53:16.600524	159.226.45.3	4090	MY.NET.6.7	25
10.04	09:53:19.723006	159.226.45.3	4090	MY.NET.6.7	25
10.04	10:20:22.955797	159.226.45.3	4107	MY.NET.6.7	25
10.04	10:20:24.398674	159.226.45.3	4107	MY.NET.6.7	25
10.04	10:20:26.055432	159.226.45.3	4107	MY.NET.6.7	25
10.04	10:24:11.135669	159.226.45.3	113	MY.NET.6.7	22690
10.04	10:24:15.623809	159.226.45.3	4109	MY.NET.6.7	25

10.04	10:24:15.630292	159.226.45.3	4109	MY.NET.6.7	25
10.04	10:24:15.651689	159.226.45.3	4109	MY.NET.6.7	25

This trace seems to indicate either missing data or a response with no stimulus. Cause for worry.

10.27	21:43:23.871929	159.226.45.3	4580	MY.NET.253.41	25
10.27	21:43:27.559680	159.226.45.3	4580	MY.NET.253.41	25
11.06	19:40:20.307242	159.226.45.3	113	MY.NET.253.41	35505
11.13	08:40:54.906789	159.226.45.3	2587	MY.NET.253.41	25
11.13	08:41:09.776421	159.226.45.3	2587	MY.NET.253.41	25

A further sign that 159.226.45.3 may not have your best interest at heart.

10.06	00:00:02.864385	159.226.45.3	1201	MY.NET.6.7	23
10.06	00:00:03.582799	159.226.45.3	1201	MY.NET.6.7	23

These captures indicate a probable compromise of MY.NET.6.7 and MY.NET.253.41. I urge you to immediately remove them from service and perform forensics to determine the extent to which a compromise has occurred. Given the intense interest in your mail servers, it would also be advisable to check all mail servers and ensure they are patched to the latest revisions.

212.179.95.5 – Alerts generated as a result of Watchlist 000220 IL-ISDNNET-990517. Source shows a focused interest in Ports around the 4000 range. Troublesome here is the targeting of the same machines over a period of several days.

10.08	12:03:27.733622	212.179.95.5	1323	MY.NET.204.34	4171
10.08	12:03:30.091801	212.179.95.5	1323	MY.NET.204.34	4171
11.10	10:51:56.932695	212.179.95.5	1983	MY.NET.204.34	4619
11.10	10:51:57.650383	212.179.95.5	1983	MY.NET.204.34	4619
10.05	17:04:11.938950	212.179.95.5	1078	MY.NET.209.202	4681
10.05	17:04:29.115771	212.179.95.5	1078	MY.NET.209.202	4681
10.05	17:04:46.586460	212.179.95.5	1078	MY.NET.209.202	4681
10.06	03:29:28.010462	212.179.95.5	1893	MY.NET.209.202	4681
10.06	03:29:28.601917	212.179.95.5	1893	MY.NET.209.202	4681
11.04	06:53:58.388774	212.179.95.5	2012	MY.NET.211.146	4922
11.04	15:19:39.764305	212.179.95.5	3288	MY.NET.211.146	4922
11.05	04:47:27.303528	212.179.95.5	1263	MY.NET.211.146	4922
11.05	04:47:28.884431	212.179.95.5	1263	MY.NET.211.146	4922
11.05	04:47:29.265523	212.179.95.5	1263	MY.NET.211.146	4922
11.05	06:59:39.204067	212.179.95.5	2270	MY.NET.222.194	4968
11.05	07:21:16.737559	212.179.95.5	2514	MY.NET.222.194	4968
11.11	04:39:26.400278	212.179.95.5	4179	MY.NET.222.194	4470
11.11	04:57:06.853051	212.179.95.5	4376	MY.NET.222.194	4470

Such activity doesn't constitute proof of a compromise directly, but considering the source, (Israel) any machines that generate repeat visits from a single hostile source warrant investigation. Such behavior by the attacker could indicate a compromise already in place or a focusing of interest to evaluate an attack mode. As such, I recommend the four targeted machines listed in the above table to be examined for signs of compromise

209.92.40.32 – This hostile has an intense interest in port 9704. He performed a large port scan over several of your class c address space networks. It is interesting to note that the tool in use also consistently originates from port 9704. I discussed the significance of port 9704 earlier, but to recap - the attacker is looking for nodes that may have already been compromised.

ALERT FILE STATISTICS – DESTINATION IP

Snort identified 26.675 unique destination addresses receiving 110,457 packets. The following table enumerates the top targets. This table also represents the top targets that are responsible for 24.4% of all Scan packets captured (27,000 packets).

Target	Events
MY.NET.6.7	5800
MY.NET.211.146	4814
MY.NET.223.98	3940
MY.NET.206.90	3918
MY.NET.70.255	1813
MY.NET.203.142	1640
MY.NET.218.142	1463
MY.NET.214.170	1371
MY.NET.100.230	1289
MY.NET.202.22	952

Top Alert Targets

MY.NET.6.7 – It's not surprising that this is the top destination, as we have already identified it as compromised. Aside from 159.226.45.3, we also see 159.226.66.130 abusing it.

10.31	09:41:36.079666	159.226.66.130	2485	MY.NET.6.7	25
10.31	09:42:09.520146	159.226.66.130	2485	MY.NET.6.7	25
10.31	09:42:09.523827	159.226.66.130	2485	MY.NET.6.7	25
10.31	09:48:00.222312	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:02.091291	159.226.66.130	113	MY.NET.6.7	22346
10.31	09:48:03.024653	159.226.66.130	113	MY.NET.6.7	22346
10.31	09:48:07.183521	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:07.191686	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:07.645385	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:09.298055	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:09.298900	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:10.134116	159.226.66.130	2518	MY.NET.6.7	25
10.31	09:48:10.136753	159.226.66.130	2518	MY.NET.6.7	25
11.07	02:32:27.573495	159.226.66.130	3717	MY.NET.6.7	25
11.07	20:36:26.091933	159.226.66.130	113	MY.NET.6.7	15276
11.07	20:36:31.544390	159.226.66.130	4701	MY.NET.6.7	25

In fact, this target was of interest to several nodes from the 159.226 network. I see numerous probes to the INETD, TELNET, and SMTP services. Snort also captured some NMAP TCP Pings (4), SYN-FIN scans (looking for DNS and port 9704), and a WinGate 1080 attempt.

MY.NET.211.146 – Of the 4814 alerts generated, 4810 of them were Watchlist from an Israeli network IP of 212.179.95.5. I'm almost positive this node is compromised, the entire data set had conversation written all over it.

11.05	05:39:56.657667	212.179.95.5	1192	MY.NET.211.146	4922
11.05	05:40:32.165296	212.179.95.5	1192	MY.NET.211.146	4922
11.05	05:40:32.944438	212.179.95.5	1192	MY.NET.211.146	4922
11.05	05:40:36.623651	212.179.95.5	1192	MY.NET.211.146	4922

11.05	05:40:39.642676	212.179.95.5	1192	MY.NET.211.146	4922
11.05	05:40:40.314472	212.179.95.5	1192	MY.NET.211.146	4922
11.05	05:40:41.954334	212.179.95.5	1192	MY.NET.211.146	4922
11.05	05:40:42.380629	212.179.95.5	1192	MY.NET.211.146	4922

.
.
.
Many more – source ports changed – but always as if a new connection was established (different times)

Nice steady source ports, 1 target, nice steady destination ports. The Destination port did bother me though, I could not find any Trojans that operated on 4922 – doesn't mean much though – NetCat can be configured to live on any port – as many Trojans can. Digging deeper, I brought up the OOS file on the Target node and found this;

11.04 02:10:05.588750 133.46.212.81:1867 MY.NET.211.146:4922
TCP TTL:110 TOS:0x0 ID:19543 DF
2*SF*PA* Seq: 0xD58F30 Ack: 0x50315 Win: 0x5B4
00 D5 8F 30 00 05 03 15 1A 5B 05 B4 6C 94 16 3A ...0.....[..l..:
00 00 00 00 00 00 ......

Actually, the OOS file had 7 entries like the above but spread out over two days November 4th and 7th. The rub here is that the source did not match. Doing an Arin on the new source I get –

Japan Network Information Center (NETBLK-JAPANB-INET)
Fuundo Bldg. 3F, 1-2 Kanda-Ogawamachi, Chiyoda-ku
Tokyo, 101-0052
JP

I pulled all of the traffic from the Scan files from this new source –

11.04 00:04:48 133.46.212.81:0 MY.NET.211.146:3827 NULL 2******* RESERVEDBITS
11.04 00:07:09 133.46.212.81:17 MY.NET.211.146:3827 NOACK 2*S*R*** RESERVEDBITS
11.04 00:07:19 133.46.212.81:3827 MY.NET.211.146:4922 NOACK 21*FR**U RESERVEDBITS
11.04 00:08:38 133.46.212.81:3827 MY.NET.211.146:4922 NOACK *1SFRP** RESERVEDBITS
11.04 01:31:23 133.46.212.81:4940 MY.NET.211.146:4922 INVALIDACK 21S*R*AU RESERVEDBITS
11.04 01:32:16 133.46.212.81:4940 MY.NET.211.146:4922 NOACK **SFRP*U
11.04 01:39:23 133.46.212.81:4940 MY.NET.211.146:4922 INVALIDACK *1S*RPA* RESERVEDBITS
11.04 01:39:49 133.46.212.81:4940 MY.NET.211.146:4922 XMAS ***F*P*U
11.04 01:46:12 133.46.212.81:0 MY.NET.211.146:4940 UNKNOWN 2*S***A* RESERVEDBITS
11.04 01:47:16 133.46.212.81:4940 MY.NET.211.146:4922 **NMAPID** 21SF*P*U RESERVEDBITS
11.04 01:58:56 133.46.212.81:0 MY.NET.211.146:1738 SYNFIN 21SF**** RESERVEDBITS
11.04 01:59:00 133.46.212.81:28 MY.NET.211.146:1738 SYN 21S***** RESERVEDBITS
11.04 02:00:10 133.46.212.81:1738 MY.NET.211.146:4922 FIN ***F****
11.04 02:10:05 133.46.212.81:1867 MY.NET.211.146:4922 INVALIDACK 2*SF*PA* RESERVEDBITS
11.04 02:13:00 133.46.212.81:0 MY.NET.211.146:1867 NOACK 21S**P** RESERVEDBITS
11.04 02:13:07 133.46.212.81:227 MY.NET.211.146:1867 UNKNOWN 2*****AU RESERVEDBITS
11.04 02:14:06 133.46.212.81:1867 MY.NET.211.146:4922 VECNA 2**F***U RESERVEDBITS

(time wise – the Israeli had his conversation here on the 4th and 5th)

11.07 01:09:51 133.46.212.81:1967 MY.NET.211.146:4922 NULL ********
11.07 02:41:56 133.46.212.81:0 MY.NET.226.234:2720 NOACK ****RP**
11.07 03:45:51 133.46.212.81:2931 MY.NET.211.146:4922 NOACK 21SFRP** RESERVEDBITS
11.07 03:45:52 133.46.212.81:2931 MY.NET.211.146:4922 NULL ********
11.07 03:45:54 133.46.212.81:2931 MY.NET.211.146:4922 NOACK 21*FRP** RESERVEDBITS

11.07 05:15:32 133.46.212.81:3706 MY.NET.211.146:4922 VECNA 21*F*P** RESERVEDBITS
11.07 05:17:42 133.46.212.81:3706 MY.NET.211.146:4922 VECNA ***F*P**
11.07 05:18:35 133.46.212.81:33 MY.NET.211.146:3706 INVALIDACK *1S*R*AU RESERVEDBITS
11.07 05:19:37 133.46.212.81:3706 MY.NET.211.146:4922 UNKNOWN 2*S***A* RESERVEDBITS
11.07 05:23:58 133.46.212.81:1 MY.NET.211.146:3706 NOACK **S*R***
11.07 05:24:11 133.46.212.81:3706 MY.NET.211.146:4922 NMAPID 2*SF*P*U RESERVEDBITS
11.07 05:24:39 133.46.212.81:3706 MY.NET.211.146:4922 UNKNOWN *1*F*PA* RESERVEDBITS
11.07 05:24:53 133.46.212.81:3706 MY.NET.211.146:4922 UNKNOWN 2**F*PA* RESERVEDBITS
11.07 05:25:28 133.46.212.81:3706 MY.NET.211.146:4922 UNKNOWN 21***PAU RESERVEDBITS

Amazing coincidence that our new friend is only interested in the same node. He even pays more attention to the same destination port as our Israeli friend. On a roll; pulled all alert traffic from our new friend;

11.04 01:59:00.495308 Queso fingerprint 133.46.212.81:28 MY.NET.211.146:1738
11.04 00:19:14.512507 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 00:20:31.785985 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 00:20:37.612343 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 00:21:25.335973 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 01:47:11.295291 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 01:47:39.181313 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 01:51:27.336537 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 01:51:40.516141 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:02:07.999994 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:02:43.866690 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:10:18.758219 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:2 UDP:0)
11.04 02:16:18.574010 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:22:15.005062 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:24:31.999867 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:24:38.133992 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.04 02:25:29.972795 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 01:09:51.890196 Null scan!_ 133.46.212.81:1967 MY.NET.211.146:4922
11.07 01:21:52.342155 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 02:53:25.462996 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 03:45:52.394024 Null scan!_ 133.46.212.81:2931 MY.NET.211.146:4922
11.07 04:02:03.607680 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:3 UDP:0)
11.07 05:31:45.095548 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:33:12.329296 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:33:44.219503 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:34:21.200874 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:36:52.772828 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:37:00.179393 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:37:16.762068 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:37:25.307981 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.07 05:37:53.885072 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.09 21:35:51.111402 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.09 22:35:44.538585 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)
11.09 22:38:47.868532 Portscan 133.46.212.81 (TOTAL HOSTS:1 TCP:1 UDP:0)

Queso fingerprint and then a Portscan! And how about those two NULL scans.

Even though I don't see the actual exploit, there is too much circumstantial evidence to not call this node as compromised. In the arena of Network Intrusion – Compromised until proven not compromised.

MY.NET.211.146

MY.NET.223.98 – Of the 3940 alerts captured for this target IP, 3938 of them were Watchlist from Israel. All traffic was targeted to port 6699 which is commonly used by Napster – a music file sharing service. Even though I'm not happy with the source, I'm inclined to call this traffic as nominal. It is important for you to note that while a machine is part of the Napster network, it is acting as a file sharing device. Although I'm aware of no direct vulnerabilities with Napster, Trojans abound on the Internet. Such services can represent an extreme threat on your network.

MY.NET.206.90 – and then - MY.NET.203.142 - While examining the data for this first node, I was struck by the popularity of destination port 4619. I was even more concerned by the fact that port 4619 was of intense interest by two separate IP's in the 212.179 domain (Israel – 212.179.27.6 & 212.179.95.5). Troubling still, this node was Null Scanned (port 4619) by 212.89.31.158 (Spain). I searched all of the Alert data (for destination port 4619) and found that still another Israeli IP, 212.179.79.2 was touching Port 4619 on node MY.NET.203.142 – which happens to be another Top Alert Destination. In fact, Port 4619 was the number two targeted port (5713 packets) for Alert (Watchlist 000220 IL-ISDNNET-990517) destinations. Such strong correlation makes me nervous. Even though I have no clues as to what is running on 4619, I'm going to call these two nodes as compromised.

MY.NET.206.90
MY.NET.203.142

MY.NET.70.255 – There are only two explanations for someone sending this into your network. Either they are doing recognizance or they are using you to attack an innocent node. Consider the following sample data set.

10.14	19:52:30.458412	193.226.127.19	MY.NET.70.255
10.14	19:53:28.474465	193.226.127.19	MY.NET.70.255
10.14	19:53:41.428002	193.226.127.19	MY.NET.70.255
10.14	19:53:47.933745	193.226.127.19	MY.NET.70.255
10.14	19:54:13.947020	193.226.127.19	MY.NET.70.255
10.14	19:54:20.405936	193.226.127.19	MY.NET.70.255
10.14	19:54:46.358268	193.226.127.19	MY.NET.70.255
10.14	19:56:23.424439	193.226.127.19	MY.NET.70.255
10.14	19:58:07.158278	193.226.127.19	MY.NET.70.255
10.14	19:58:59.020213	193.226.127.19	MY.NET.70.255
10.14	19:59:24.909340	193.226.127.19	MY.NET.70.255
10.14	19:59:50.814552	193.226.127.19	MY.NET.70.255
10.14	19:59:57.312846	193.226.127.19	MY.NET.70.255
10.14	20:00:10.297044	193.226.127.19	MY.NET.70.255
10.18	16:12:25.860722	193.226.127.19	MY.NET.70.255
10.18	16:12:38.786875	193.226.127.19	MY.NET.70.255
10.18	16:13:37.152863	193.226.127.19	MY.NET.70.255
10.18	16:13:43.615427	193.226.127.19	MY.NET.70.255
10.18	16:14:03.040003	193.226.127.19	MY.NET.70.255
10.13	18:30:01.785973	193.226.127.20	MY.NET.70.255
10.13	18:30:40.716682	193.226.127.20	MY.NET.70.255
10.13	18:32:04.846814	193.226.127.20	MY.NET.70.255
10.13	18:34:21.055037	193.226.127.20	MY.NET.70.255
10.13	18:39:25.743134	193.226.127.20	MY.NET.70.255
10.13	18:42:14.125192	193.226.127.20	MY.NET.70.255
10.13	18:42:20.506605	193.226.127.20	MY.NET.70.255
10.13	18:43:25.429707	193.226.127.20	MY.NET.70.255

10.13	18:49:02.635672	193.226.127.20	MY.NET.70.255
10.13	18:49:47.881839	193.226.127.20	MY.NET.70.255
10.14	19:18:55.242474	193.226.127.20	MY.NET.70.255
10.14	19:21:04.865687	193.226.127.20	MY.NET.70.255
10.14	19:21:43.802411	193.226.127.20	MY.NET.70.255
10.14	19:22:42.253135	193.226.127.20	MY.NET.70.255
10.14	19:24:19.990280	193.226.127.20	MY.NET.70.255
10.14	19:24:45.625246	193.226.127.20	MY.NET.70.255
10.14	19:25:24.443898	193.226.127.20	MY.NET.70.255
10.14	19:30:54.994143	193.226.127.20	MY.NET.70.255
10.14	19:32:58.146537	193.226.127.20	MY.NET.70.255
10.11	17:18:37.973340	193.226.127.21	MY.NET.70.255
10.11	17:54:47.404686	193.226.127.21	MY.NET.70.255
10.19	16:32:55.227203	193.226.127.21	MY.NET.70.255
10.19	16:34:26.225768	193.226.127.21	MY.NET.70.255
10.19	16:36:22.591223	193.226.127.21	MY.NET.70.255

Here we see multiple broadcast pings to three consecutive addresses on the 193.226.127 network. I would bet that someone is spoofing the source address in order to launch a Denial of Service attack on them.

Conversely, examine this single broadcast ping.

10.27	23:00:12.996638	193.226.181.57	MY.NET.70.255

It was the only packet from this class c – although it could be part of a collaborative attempt to launch a DoS against the source IP, I suspect that it is being utilize to map your MY.NET.70 subnet. You are being badly abused here. I counted 216 unique sources throwing broadcast pings into your network. No doubt you are very popular throughout the script kiddie community. I suggest configuring your border routers to block broadcast pings both in and out of your network.

MY.NET.218.142 – Interesting, we find a similar circumstance that we saw in the previous analysis of MY.NET.206.90 and MY.NET.203.142. The difference is that this time it is a single conversation (constant source port) going to port 4990 from node 212.179.79.2. The overall amount of traffic going to port 4990 seems nominal – no *bright* red flags. However, considering the source is from a hostile network and the destination port is a little higher than I like, I'm going to call this one as something to check out for signs of compromise.

MY.NET.218.142

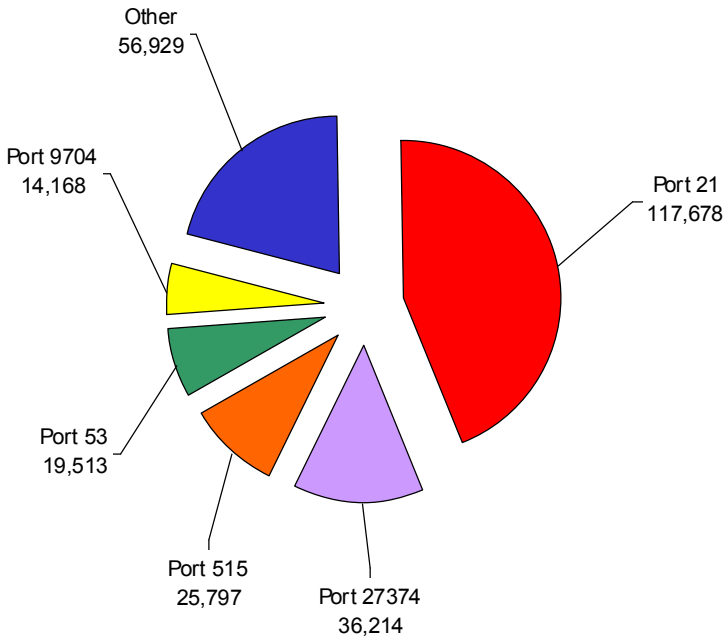
MY.NET.214.170 – Lots of connections to Port 6699, a good sign that we found another user enjoying some afternoon music courtesy of Napster. This node also received a number of WinGate 1080 attempts (16). As such, examine this machine for WinGate and ensure that it is configured to not allow outbound connections.

MY.NET.100.230 – Source IP 159.226.91.20 has a real interest in your SMTP service (port 25). Considering we have already seen machines from this network compromise your MY.NET.6.7 node, this is not so surprising. It would seem that the Chinese have a keen interest in your mail servers. Source IP 159.226.63.200 (China) also tried to perform some recon on your IDENT (port 113) service. This service can give up significant amounts of information, you may want to consider turning it off. Considering their apparent focused interest in your mail servers, I suggest checking over all of your mail servers for compromise and ensure that the operating system and services are patched to the most current levels.

MY.NET.202.22 – Another Napster user. The fact that three of the top ten Alert Destinations have been Napster related should give you a clear picture on how much of your network resources (bandwidth) are being wasted on non business related activities. If you really want to identify all the nodes sending or receiving Napster traffic, please drop me an email and I will dump them for you.

Scan Destination Port Analysis

All of the scan data files were sorted by destination port. The following are the results:



Port 21 - Clearly FTP is a favorite. I would suggest that you evaluate this service on your network. Disable the service if not needed, ensure the software is up to date on all boxes that require it.

Port 27374 – Another favorite of the underground. This is the service port for a program called SubSeven. It is a wicked little program allowing the Cracker full access to the infected node. Not surprisingly, most of the probes are coming from the 24 network – the province of cable modems and script kiddies. Key to the technology of SubSeven (and most all other Trojans for that matter) is the fact they can be run by individuals who can't spell IP. Your best bet for defense against them are to ensure you have good up to date antiviral protection on all desktops and servers.

Port 515 – Unix LPR service. As covered earlier, this was a newly discovered vulnerability during this time frame.

Port 53 – DNS No surprise here, DNS vulnerabilities are numerous and when compromised usually wreak havoc on the network. Key issues here are patching the OS and service to current levels, use a split DNS configuration to protect the internal network , and limit zone transfers to approved DNS servers.

Port 9704 – New Trojan As already covered, this indicates a hostile intent by looking for a target already compromised with a wu-ftpd exploit or inetd exploit.

Watchlist Source Ports

There are a few ports that we do not want to see as source ports from unfriendly networks. One of them is the IDENT port (113). The following nodes have apparently queried the source IP on port 113 in response to a connection.

MY.NET.6.47
MY.NET.110.150
MY.NET.253.43

Data from the Watchlist 000222 NET-NCFC report

Interesting, all three were targeted by the same IP address (212.179.39.194 - clnt-39194.bezeqint.net – our Israeli friends.) and all targeted nodes appear to be running SMTP services.

MY.NET.6.7
MY.NET.6.47
MY.NET.253.43
MY.NET.253.42
MY.NET.253.41
MY.NET.145.9
MY.NET.110.150
MY.NET.100.230

Data from the Watchlist 000222 NET NCFC report

As previously suspected, the Computer Network Center Chinese Academy of Sciences has a deep interest in your mail servers.

I reiterate at this time my position on these alerts. The single fact that your mail servers queried port 113 does not constitute a compromise of your mail servers. However, considering the sources are known to be hostile and email servers are a critical network service, I will err on the side of caution and place these in the compromised category.

Alerts by Snort Rule

Snort generates alerts in response to pattern matching in which false positives are rare. It is therefore safe to say that all alerts represent some level of threat to your unprotected network. With few exceptions, most are reconnaissance in nature and can be blocked at a border router or firewall – so there is little benefit in analyzing each and every log individually. Instead, we will aggregate the information and use this data to examine what is popular in terms of activity, as well as identify who the top talkers and top receivers are.

First, here is a list of all alerts captured via the Snort logs and their 'hit counts'.

SYN-FIN Scan	56250
Watchlist220	30997
Portscan	27118
Watchlist222	8134
WinGate1080	4764
TCP SMTP Source Port traffic	2893
Attempted Sun RPC high port access	2542
Broadcast Ping to subnet	1813
Back Orifice	1697
SNMP public access	468
Null Scan	277
SMB Name Wildcard	218
Queso fingerprint	142
NMAP TCP Ping	96
SUNRPC Highport access	60
Connect to 515 from inside	56
Probable NMAP fingerprint attempt.txt	15
External RPC call	13
Possible wuftpd exploit	13
Tiny Fragments	7
Happy 99 Virus	2
Total events captured	**150328**

SYN-FIN / NULL Scans

As you can see, the SYN-FIN scan represents over a third of all alert events. With its brother the NULL scan, they comprise an attempt to footprint a network to ascertain what assets can be reached. It is a very popular method in mapping a network; as evident by its position on the above chart as well as the fact that Snort captured 30 different external IP addresses employing a SYN-FIN scan and 204 different external IP addresses employing a NULL scan against you.

Watchlist 220 & 222

Separately, these reside as numbers 2 and 4 on the list, combined they represent almost 20% of the total amount of alerts generated by Snort. Simply, Watchlist alerts indicate traffic from an IP address (subnet) that has been classified as unfriendly. For the alerts that Snort has currently trapped, we see Watchlist alerts from the 212.179 network and the 159.226 network. This is of concern to us since the 212.179 block is originating from Israel and the 159.226 block is originating from China. Such sources should be a major concern especially if your organization does not do business in those geographical areas.

Port Scans

Port scans represent another large chuck of activity targeted at your network. Rather than footprinting as we've seen with SYN-FIN scans, these probes are designed to target specific hosts and test for what

services are present. The unfortunate reality of a Port Scan is that it generally means your network was already mapped to some extent and now the source is looking for a service to compromise.

WinGate1080, Possible wuftpd exploit, Attempted Sun RPC high port access, SUNRPC Highport access, External RPC call, SMB Name Wildcard, & TCP SMTP Source Port traffic
We loosely group these together since they are all variations on the same theme. The attacker is attempting to query / exploit specific services that are running on your network.

WinGate 1080 – This alert is not a compromise per-se. The source IP address is looking for a poorly configured WinGate server so that he or she may 'bounce' through your nodes and target other systems. Such activity hides the attacker's IP address from the target's logs and is considered to be quite the jewel – as evident by the 569 unique source IP addresses captured by Snort. It may also be an indication that nodes on your network are trying to communicate with IRC chat servers. Many IRC chat servers will scan the IRC user's machine for the presence of Socks (port 1080) to prevent users from connecting through a 'bounced' WinGate. Given your current state of security, I would expect to find evidence of both on your network.

Remedy – If you are allowing WinGate machines on your network (inadvisable) ensure they are configured to deny external inbound connections. If your company policies disallow ICQ chatting, identify and remove the service from affected nodes.

Ref CERT® Vulnerability Note VN-98.03 WinGate IP Laundering

Possible wuftpd exploit – FTP provides file transfer services between computers. The "flavor" of FTP known as WUFTP (Washington University FTP Server) is known to suffer from several buffer overflow vulnerabilities that allow an attacker to attain administrator access to the server. Snort has detected exploit code being directed towards FTP services on your network.

Remedy – The following computers were targeted with the exploit code. Examine each and ensure that the FTP service is patched to the latest revision level. If you are running a WUFTP version prior to version 2.6.0, you are almost certainly compromised. You may want to evaluate if you need to be running FTP services at all.

Ref CERT® Advisory CA-1995-16 wu-ftpd Misconfiguration Vulnerability
Original issue date: November 30, 1995
CERT® Advisory CA-1999-13 Multiple Vulnerabilities in WU-FTPD
Original release date: October 19, 1999

MY.NET.130.242
MY.NET.130.81
MY.NET.205.94
MY.NET.221.82
MY.NET.97.206
MY.NET.99.130

External RPC call (portmapper 111)
Attempted Sun RPC high port access (32771)
SUNRPC High Port Access (32771) – Unlike many of the other Snort Alerts, we see the potential for a lot of false positives concerning the SUNRPC High Port Access. Unfortunately, the ports in question is also used by AOL's ICQ service. As such, we need to 'filter out' the noise to see if we have any noteworthy port 32771 activity.

The following machines were resolved through ARIN lookups to be ICQ servers on AOL;

205.188.153.97	205.188.153.107

205.188.153.98	205.188.153.108
205.188.153.99	205.188.153.109
205.188.153.100	205.188.153.110
205.188.153.101	205.188.153.111
205.188.153.102	205.188.153.114
205.188.153.104	205.188.153.115
205.188.153.105	205.188.153.116
205.188.153.106	

Since the above names do resolve to AOL ICQ servers, I feel comfortable that the following machines were not being targeted for RPC enumeration / attack. It is also noteworthy to mention that RPC attacks are only directed at UNIX or LINUX boxes, therefore if any of the following boxes are running a Microsoft operating system, it would be further evidence that rather than an attack, we have some bored employees chatting over the Internet. The following internal IP's are suspected of participating in ICQ chats with one or more of the above listed ICQ servers. Your company policy may or may not allow this activity.

MY.NET.97.62	MY.NET.220.194
MY.NET.97.152	MY.NET.221.126
MY.NET.97.163	MY.NET.221.256
MY.NET.97.202	MY.NET.222.98
MY.NET.98.190	MY.NET.223.18
MY.NET.105.115	MY.NET.224.214
MY.NET.144.42	MY.NET.225.98
MY.NET.152.198	MY.NET.225.106
MY.NET.202.242	MY.NET.225.210
MY.NET.206.222	MY.NET.226.74
MY.NET.209.182	MY.NET.226.198
MY.NET.217.214	MY.NET.227.50
MY.NET.217.218	MY.NET.227.170
MY.NET.219.130	MY.NET.228.22
MY.NET.220.78	MY.NET.228.42

I also uncovered some evidence that some users may be performing online banking and investing. The node MY.NET.6.15 has communicated with Laurel Web Online Services and Belz Investment Company. Nodes My.NET.15.27 and MY.NET.100.130 both communicated with Belz Investment company.

The remainder of the machines may be another story. Gathering information from various sources, (notably ARIN & RIPE) I produced a who's who list of source IP addresses that you do not want to have knocking on your door;

- Office of Computer Services at Utah State
- Russia
- Germany
- Brazil – The infamous Brazilian Research Network
- Korea
- UUNET
- Cable Modem users (ATHOME)
- Darkorb Communications in Wilmington Delaware. This was not immediately recognized as a problem child. However, given the fact that they fell into 'top talker' status, I decided to do some further digging. I found their upstream provider was Kinetic.Cpanel.net. I became more than a little concerned when I saw that their DNS servers were named;

REVOLT.DARKORB.NET
DESTRUCTION.DARKORB.NET.

Remedy - Given the observed source addresses, I believe it to be prudent to check these machines for signs of compromise. I reiterate that these probes against port 32771 are interested in UNIX or LINUX machines only.

MY.NET.1.8	MY.NET.206.222
MY.NET.100.130	MY.NET.212.186
MY.NET.140.51	MY.NET.228.62
MY.NET.179.78	MY.NET.253.114
MY.NET.202.242	MY.NET.53.14
MY.NET.204.134	MY.NET.53.23
MY.NET.205.130	MY.NET.6.15
MY.NET.206.218	MY.NET.97.59

RPC High Port Targets

Likewise, these three machines were queried for Portmapper (111) services, they should also be checked and if possible portmapper should be disabled.

MY.NET.100.130
MY.NET.15.27
MY.NET.6.15

Portmapper Target

SMB Name Wildcard
SMB traffic was observed between several internal network nodes. This is to be expected. The concern is SMB traffic flowing out of your network. (16 different source IP's were captured potentially exchanging data over port 137)

Remedy - You need to disallow ports 135, 137 and 139 at your firewall to prevent the hemorrhage of information – or data – from your network.

TCP SMTP Source Port traffic
This looks to be several scans for the mail service. Most SMTP scans are poking around for open relays to facilitate spamming. Another possibility is the ever present script kiddie looking for a node to compromise.

Remedy – Configure your mail servers to not act as mail relays. Block inbound port 25 traffic except to designated mail servers.

Broadcast Ping to subnet
This was already covered earlier.

Back Orifice Ping
Back Orifice is a Trojan used to remote a node. What I see in the traces are several sweeps of your address space looking for the 'pong' to the pinging of port 31337.

Remedy - Like the SubSeven probes earlier, the best way to protect against a Trojan is to have up to date virus protection on all servers and nodes.

SNMP public access
SNMP is capable of giving up vast amounts of information about your network infrastructure. The good news is that no one outside of your network has discovered that you are still using the default community

string of 'public'. Snort came across this by capturing nodes on your MY.NET.97 network talking SNMP to host MY.NET.101.192.

Remedy – Evaluate the need for SNMP and if not needed turn the service off. If SNMP is operationally necessary, change the default community string.

Queso fingerprint
NMAP TCP Ping
Probable NMAP fingerprint attempt
Queso and Nmap are two programs commonly used to fingerprint the TCP stack on the targets. This information can then be used to decide which exploits to launch.

Remedy – All such fingerprinting, as well as scanning can be blocked by a firewall.

Connect to 515 from inside
We covered why port 515 is significant earlier. What is noteworthy here is that some of the scans are coming from inside of your network. I took a closer look at the source IP and found 2 nodes as sources, MY.NET.101.142 and MY.NET.179.78.

11.22	11:24:06.406682	MY.NET.179.78	2274	64.244.202.110	515
11.22	11:33:56.296324	MY.NET.179.78	2707	64.244.202.66	515

The 142 node seems to be clean – no other entries in any other list. In fact I'm inclined to believe the 515 problem is a print que acting up. However, the .78 node also shows up in the Alert logs as:

09.28 13:28:03.304676 SUNRPC highport access!_ 24.18.90.197:4795 MY.NET.179.78:32771

I analyzed this earlier and suspected there might be a problem since the 24 address does not look to be a ICQ server (cc53440-a.catv1.md.home.com – cable modem). The other significant characteristic is that your internal node is scanning two computers on the 64.244.202 network. Very bad.

64.244.202.66 igw.healthcite.com
64.244.202.110 mail.healthcite.com

Given this evidence, I will place node MY.NET.179.78 on the list for scrutiny.

MY.NET.179.78

Tiny Fragments
Take off on a Don Ho song. Sorry, long night.
The fragmentation of TCP packets is often used to circumvent a packet filtering device or IDS system. As such, they are not used to exploit target nodes although the payload can contain exploit code.

Happy 99 Virus
Snort detected this worm heading towards your email server.

Remedy – Use a antiviral protection on your email server (such as Scanmail).

Assignment 3 – Analysis Process

After spending several days just staring at it, I came to the conclusion that I needed to separate the day out to make it manageable. Call me old fashioned but what immediately came to mind is a fast computer and some batch files. Sorry, I didn't feel the need to brush off my Perl skills. <shrug> My goal was to break the data out in the following categories;

- SCAN FILES
  - By Source IP
  - By Source Port
  - By Destination IP
  - By Destination Port
- Alert Files
  - By Source IP
  - By Source Port
  - By Destination IP
  - By Destination Port
  - Group by Alert
- OOS Files
  - By Source IP
  - By Source Port
  - By Destination IP
  - By Destination Port

Of course there were some problems to overcome first. Batch files don't have the best string handling features in the word, so first I needed to format the files so they were consistent in what was contained in column one, two, etc. etc. Instrumental in this file, as well as others, is a program called T.EXE. This program has some absolutely wonderful string handling features that are utilized through pipes. Please see Appendix H for a screen dump of it's help file. (MS users no longer need to be jealous of GREP.)

Appendix A contains the code for processing the Scan Files
Appendix B contains the code for processing the Alert Files
Appendix C contains the code for processing the OOS Files

From there, I used these files to break each type of file into by source IP and by destination IP

Appendix D contains the code for breaking the scan files
Appendix E contains the code for breaking the Alert files
Appendix F contains the code for breaking the OSS files

One more breakdown – by alert type –

Appendix G contains the code for breaking the Alert files down by alert

From this point, I used several more small batch files to break down the data by source and destination port. Mostly they were bastardizations of the code for breaking down the IP addresses into source and destination. Unfortunately, they are archaic to look at, I didn't have time to pretty them up. (current time – 3.5 hours till deadline – and I still need to spell check this pig)

I then set the mess in motion. It took a Dell Power Edge 2450 server 2 solid days to crunch it all. Batch files are very slow. The end result was about 250MB of data in 96,432 files. <chuckle> Not pretty – but it was sorted every which way.

As you can see by assignment 2, I took the top talker approach. My cut off point was either the top ten talkers for their class, or the top 25% (in packets sent or received). After analyzing them, I moved to the 'by alert type' files and took each in turn. By this time, I was seeing some patterns concerning ports (source and destination) and zeroed in on some specific ones for analysis.

By breaking down all the data such as I did, it was a simple means to perform analysis on individual files in Excel using the sort and filter functions.

Name look ups were accomplished with Sam Spade – In my humble opinion, the perfect tool to perform this kind of work.

Were there a few things I would do differently? Yes – I found myself constantly getting tied up in non compromised events like Napster or ICQ. It was a real effort to try and stay on track looking for bad things. I also tended to spend a lot of time looking for collaboration among the hostile – never found any but for whatever reason I was convinced they were there. As a training exercise, this was very good. It shattered my notion of black and white. I spent many hours looking for the obvious compromises only to realize it is a much more subjected process. Interesting, it is a concept that you would never learn from a regular test.

Appendix A – InRawScan.cmd

This file took the raw snort scan files, formatted them to a 'standard' and renamed them according to date.

```
:: Program by John Topp

@echo off
cls
title Processing Raw Scan files \RawScanData - .\DataInScan
if not exist .\DataInScan mkdir .\DataInScan
        :: Get list of all Raw Snort Scan files
dir .\RawScanData /b >.\DirList.tmp
        :: Process one file at a time
for /f "tokens=1" %%L in ('type .\DirList.tmp') do call :sFile %%L
        :: Count total events
set /A ETotal=0+0
for /f "tokens=2" %%M in ('type .\DataInScan\TotalEvents.wri') do call :tot %%M
echo. >> .\DataInScan\TotalEvents.wri
echo Total Events:      %ETotal% >> .\DataInScan\TotalEvents.wri
        :: Clean up
del .\DirList.tmp
del .\*.tmp /Q >nul
now InRaw.cmd Finish Run
goto :eof

        :sFile
        now.exe Processing %1
        title Processing Log %1
                :: need to remove inadvertant redirector symbol
        type .\RawScanData\%1 |T repl '-#3E ' " >.\raw1.tmp
                :: Change date field from alphanumeric to numeric
        echo Test Line Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec / . >>.\raw1.tmp

        type .\raw1.tmp |T repl 'Jan  ' '01.0' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Feb  ' '02.0' >.\raw1.tmp
        type .\raw1.tmp |T repl 'Mar  ' '03.0' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Apr  ' '04.0' >.\raw1.tmp
        type .\raw1.tmp |T repl 'May  ' '05.0' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Jun  ' '06.0' >.\raw1.tmp
        type .\raw1.tmp |T repl 'Jul  ' '07.0' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Aug  ' '08.0' >.\raw1.tmp
        type .\raw1.tmp |T repl 'Sep  ' '09.0' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Oct  ' '10.0' >.\raw1.tmp
        type .\raw1.tmp |T repl 'Nov  ' '11.0' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Dec  ' '12.0' >.\raw1.tmp

        echo Test Line Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec / . >>.\raw1.tmp

        type .\raw1.tmp |T repl 'Jan ' '01.' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Feb ' '02.' >.\raw1.tmp
        type .\raw1.tmp |T repl 'Mar ' '03.' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Apr ' '04.' >.\raw1.tmp
        type .\raw1.tmp |T repl 'May ' '05.' >.\raw2.tmp
        type .\raw2.tmp |T repl 'Jun ' '06.' >.\raw1.tmp
        type .\raw1.tmp |T repl 'Jul ' '07.' >.\raw2.tmp
```

```
type .\raw2.tmp |T repl 'Aug ' '08.' >.\raw1.tmp
type .\raw1.tmp |T repl 'Sep ' '09.' >.\raw2.tmp
type .\raw2.tmp |T repl 'Oct ' '10.' >.\raw1.tmp
type .\raw1.tmp |T repl 'Nov ' '11.' >.\raw2.tmp
type .\raw2.tmp |T repl 'Dec ' '12.' >.\raw1.tmp
type .\raw1.tmp |T repl '/ ' '/' >.\raw2.tmp
type .\raw2.tmp |T repl 'Test Line' ''>.\raw1.tmp

        :: Remove Heading
type .\raw1.tmp |T incl 'MY.NET' > .\raw2.tmp
        :: Get first line
type .\raw2.tmp |T top 1 > .\First.tmp
for /f "tokens=1" %%J in ('type .\First.tmp') do set FileNameF=%%J
        :: Get last line
type .\raw2.tmp |T bottom 1 > .\last.tmp
for /f "tokens=1" %%K in ('type .\last.tmp') do set FileNameL=%%K
        :: Copy formatted report to working directory
copy .\raw2.tmp  .\DataInScan\%FileNameF%_%FileNameL%_Scan.txt >nul
        :: Count the lines
type .\raw2.tmp |T count > .\LineCount.tmp
for /f "tokens=1" %%L in ('type .\LineCount.tmp') do call :Update %%L
goto :eof

:Update
echo %FileNameF%_%FileNameL%_Scan.txt %1 events counted
>>.\DataInScan\TotalEvents.wri
goto :eof

:tot
set /A ETotal=%ETotal%+%1
goto :eof

:eof
```

Appendix B – InRawAlert.cmd

This file took the raw snort alert files, formatted them to a 'standard' and renamed them according to date.

```
:: Program by John Topp

@echo off
cls
title Processing Raw Attack files \RawAttackData - .\DataInAttack
if not exist .\DataInAttack mkdir .\DataInAttack
        :: Get list of all Raw Snort Scan files
dir .\RawAttackData /b >.\DirList.tmp
        :: Process one file at a time
for /f "tokens=1" %%L in ('type .\DirList.tmp') do call :sFile %%L
        :: Count total events
set /A ETotal=0+0
for /f "tokens=2" %%M in ('type .\DataInAttack\TotalEvents.wri') do call :tot %%M
echo. >> .\DataInScan\TotalEvents.wri
echo Total Events:      %ETotal% >> .\DataInAttack\TotalEvents.wri
        :: Clean up
del .\DirList.tmp
del .\*.tmp /Q >nul
now InRawAttack.cmd Finish Run
goto :eof

        :sFile
        now.exe Processing %1
        title Processing Log %1
                :: need to remove inadvertant redirector symbol
        type .\RawAttackData\%1 |T repl '-#3E ' " >.\raw1.tmp
                :: Remove Heading
        type .\raw1.tmp |T incl '[**]' > .\raw2.tmp
                :: Remove useless lines
        type .\raw2.tmp |T repl ' [**] ' ' > .\raw1.tmp
                :: Separate data nd time
        type .\raw1.tmp |T ins 6 '$$$' > .\raw2.tmp
        type .\raw2.tmp |T repl '$$$-' ' ' > .\raw1.tmp
        type .\raw1.tmp |T repl '/' '.' > .\raw2.tmp

                :: Get first line
        type .\raw2.tmp |T top 1 > .\First.tmp
        for /f "tokens=1" %%J in ('type .\First.tmp') do set FileNameF=%%J
                :: Get last line
        type .\raw2.tmp |T bottom 1 > .\last.tmp
        for /f "tokens=1" %%K in ('type .\last.tmp') do set FileNameL=%%K
                :: Copy formatted report to working directory
        copy .\raw2.tmp  .\DataInAttack\%FileNameF%_%FileNameL%_Attack.txt >nul

        :: Count the lines
        type .\raw2.tmp |T count > .\LineCount.tmp
        for /f "tokens=1" %%L in ('type .\LineCount.tmp') do call :Update %%L
        goto :eof

        :Update
```

```
        echo %FileNameF%_%FileNameL%_Attack.txt %1 events counted
>>.\DataInAttack\TotalEvents.wri
        goto :eof

        :tot
        set /A ETotal=%ETotal%+%1
        goto :eof

:eof
```

Appendix C – InRawOOS.cmd

This file took the raw snort OOS files, formatted them to a 'standard' and renamed them according to date.

```
:: Program by John Topp
:: ToppJJ@state.gov

@echo off
cls
title Processing Raw OOS files RawOOSData - DataInOOS
if not exist .\DataInOOS mkdir .\DataInOOS
   :: Get list of all Raw Snort Scan files
dir .\RawOOSData /b >.\DirList.tmp
   :: Process one file at a time
for /f "tokens=1" %%L in ('type .\DirList.tmp') do call :sFile %%L

   :: Count total events
set /A ETotal=0+0
for /f "tokens=2" %%M in ('type .\DataInOOS\TotalEvents.wri') do call :tot %%M
echo. >> .\DataInOOS\TotalEvents.wri
echo Total Events:      %ETotal% >> .\DataInOOS\TotalEvents.wri
   :: Clean up
del .\DirList.tmp
del .\*.tmp /Q >nul
now InRawOOS.cmd Finish Run
goto :eof

   :sFile
   now.exe Processing %1
   title Processing Log %1
         :: need to remove inadvertant redirector symbol
   type .\RawOOSData\%1 |T repl '-#3E ' " >.\raw1.tmp
         :: Strip out header
   type .\raw1.tmp |T excl 'Subject: OOS check' >.\raw2.tmp
   type .\raw2.tmp |T excl 'Initializing Network Interface' >.\raw1.tmp
   type .\raw1.tmp |T excl 'snaplen' >.\raw2.tmp
   type .\raw2.tmp |T excl 'Entering readback' >.\raw1.tmp
   type .\raw1.tmp |T repl '-' ' ' >.\raw2.tmp
   type .\raw2.tmp |T repl '/' '.' >.\raw1.tmp
   type .\raw1.tmp |T cull 'Exiting' '==========' >.\raw2.tmp
   type .\raw2.tmp |T cull 'Snort processed' '==========' >.\raw1.tmp
   type .\raw1.tmp |T noblank  >.\raw2.tmp

         :: Get first line
   type .\raw2.tmp |T top 1 > .\First.tmp
   for /f "tokens=1" %%J in ('type .\First.tmp') do set FileNameF=%%J
         :: Copy formatted report to working directory
   copy .\raw2.tmp  .\DataInOOS\%FileNameF%_OOS.txt >nul
         :: Count the lines
   type .\raw2.tmp |T incl '=+' > .\raw1.tmp
   type .\raw1.tmp |T count > .\LineCount.tmp
   for /f "tokens=1" %%L in ('type .\LineCount.tmp') do call :Update %%L
   goto :eof

   :Update
```

```
   echo %FileNameF%_OOS.txt %1 events counted >>.\DataInOOS\TotalEvents.wri
   goto :eof

   :tot
   set /A ETotal=%ETotal%+%1
   goto :eof

:eof
```

Appendix D – SortScanIP.cmd

This file reads in the scan files and breaks each into text files that are by source IP and by destination IP

```
:: Program by John Topp

@echo off
cls
title Sorting by Source and Destination IP Address (Attack)
if not exist .\ByAttack\BySRCIP mkdir .\ByAttack\BySRCIP
if not exist .\ByAttack\ByDSTIP mkdir .\ByAttack\ByDSTIP
        :: Get List of attack files
dir .\ByAttack\*.txt /b >.\DirList.tmp
        :: Go process each file in turn
for /f "tokens=1" %%L in ('type .\DirList.tmp') do call :sAFile %%L


pause



        :: Count up the Src Scans
:: dir .\ByAttack\BySRCIP\*.txt /b >.\DirList.tmp
:: for /f "tokens=1" %%M in ('type .\DirList.tmp') do call :sSrcCount %%M
        :: Count up the DST Scans
dir .\ByAttack\ByDSTIP\s*.txt /b >.\DirList.tmp
for /f "tokens=1" %%N in ('type .\DirList.tmp') do call :sDSTCount %%N

        :: Do totals
set /A ETotal=0+0
for /f "tokens=2" %%O in ('type .\ByAttack\BySRCIP\TotalEvents.wri') do call :tot %%O
echo. >> .\ByAttack\BySRCIP\TotalEvents.wri
echo Total Events:      %ETotal% >> .\ByAttack\BySRCIP\TotalEvents.wri
set /A ETotal=0+0
for /f "tokens=2" %%P in ('type .\ByAttack\ByDSTIP\TotalEvents.wri') do call :tot %%P
echo. >> .\ByAttack\ByDSTIP\TotalEvents.wri
echo Total Events:      %ETotal% >> .\ByAttack\ByDSTIP\TotalEvents.wri
if exist *.tmp del .\*.tmp >nul
del .\CurrFile.txt >nul
if exist .\ByAttack\*.tmp del .\ByAttack\*.tmp >nul
now EndRun SortAttackIP.cmd
goto :eof

        :sAFile
                :: %1 - file to process
        set leFileName=%1
        now.exe Processing Attack File  %1
        title Processing Attack File %1
                :: read a line and sub to separate the IP from the port
        for /f "tokens=1-9" %%g in ('type .\ByAttack\%1') do call :sSocket %%g %%h %%i %%j %%k
%%l %%m %%n %%o
        goto :eof

                :sSocket
                        :: reform orginal line
```

```
                set leCurLin=%1 %2 %3 %4 %5 %6 %7 %8 %9
                set leIPSRC=%4
                set leIPTRG=%5
                        :: Break Source socket into IP and Port, go sub to update files
::              for /f "Delims=: tokens=1,2" %%I in ("%leIPSRC%") do call :sSRCIPLog %%I %%J
                        :: Break Destination socket into IP and Port, go sub to update files
                for /f "Delims=: tokens=1,2" %%I in ("%leIPTRG%") do call :sDSTIPLog %%I %%J
                echo   %1 %2 %4 %5 %6 %7 %8 %9
                goto :eof

                        :sSRCIPLog
                                :: %1 = Source IP address
                                :: %2 = Source Port

                                :: Kludge - We only need to collect src data from one of the
                                :: PortScan Alert files.
                        if /I %leFileName%==tPortscanD.txt goto :eof
                        if /I %leFileName%==tPortscanS.txt goto :eof
                                :: Stuff it
                        echo %leCurLin% >>.\ByAttack\BySRCIP\s%1.txt
                        goto :eof

                        :sDSTIPLOG
                                :: %1 = Source IP address
                                :: %2 = Source Port

                                :: Kludge - the PortScan files do not have destination IP's
                                :: skip them
                        if /I %leFileName%==tPortscanD.txt goto :eof
                        if /I %leFileName%==tPortscanS.txt goto :eof
                        if /I %leFileName%==tPortscanE.txt goto :eof

                                :: Break the IP address down to a class C and
                                :: create a directory for each
::                      echo %1 > .\env3.tmp
::                      for /f "Delims=. tokens=1,2,3" %%K in ('type .\env3.tmp') do set
NetID=%%K.%%L.%%M
::                      if not exist .\ByAttack\ByDstIP\s%NetID% mkdir
.\ByAttack\ByDstIP\s%NetID%
                                :: Now update the log
::                      echo %leCurLin% >>.\ByAttack\ByDSTIP\s%NetID%\s%1.txt
echo %leCurLin% >>.\ByAttack\ByDSTIP\s%1.txt
                        goto :eof

        :sSrcCount
        set FileName=%1
        type .\ByAttack\BySRCIP\%1 |T count > .\LineCount.tmp
        for /f "tokens=1" %%N in ('type .\LineCount.tmp') do call :UpdateSRC %%N
        goto :eof

                :UpdateSRC
                echo %FileName% %1 events counted >>.\ByAttack\BySRCIP\TotalEvents.wri
                echo %FileName% %1 events counted
                goto :eof
```

```
        :sDSTCount
                :: Process each Subdirectory (subnet)
::      set SubNetDir=%1
::      set /A SubCount=0+0
::      set /A NodeCnt=0+0
::      dir .\ByAttack\ByDSTIP\%SubNetDir%\*.txt /b >.\DirList2.tmp
::      for /f "tokens=1" %%i in ('type .\DirList2.tmp') do call :SubNetDSTCount %%i
::      set /a ratio= %SubCount% / %NodeCnt%
::      echo %SubNetDir% %SubCount% events counted for %NodeCnt% nodes (%Ratio%)
>>.\ByAttack\ByDSTIP\TotalEvents.wri
::      echo %SubNetDir% %SubCount% events counted for %NodeCnt% nodes (%Ratio%)

        set FileName=%1
        type .\ByAttack\ByDSTIP\%1 |T count > .\LineCount.tmp
        for /f "tokens=1" %%N in ('type .\LineCount.tmp') do call :UpdateDST %%N
        goto :eof

                :UpdateDST
                echo %FileName% %1 events counted >>.\ByAttack\ByDSTIP\TotalEvents.wri
                echo %FileName% %1 events counted
                goto :eof


                :SubNetDSTCount
                        :: Process each file (node) in each Subdirectory (subnet)
                set /A NodeCnt=%NodeCnt%+1
                type .\ByAttack\ByDSTIP\%SubNetDir%\%1 |T count > .\LineCount.tmp
                for /f "tokens=1" %%j in ('type .\LineCount.tmp') do call :SubTotDST %%j
                goto :eof


                        :SubTotDST
                                :: Keep a running total for each subnet
                        set /A SubCount=%SubCount%+%1
                        goto :eof


        :tot
        set /A ETotal=%ETotal%+%1
        goto :eof


:eof
```

Appendix E – SortAlertIP.cmd

This file breaks the Alert files into Source and destination text files

```
:: Program by John Topp
:: ToppJJ@state.gov

@echo off
cls
title Sorting by Source and Destination IP Address (Attack)
if not exist .\ByAttack\BySRCIP mkdir .\ByAttack\BySRCIP
if not exist .\ByAttack\ByDSTIP mkdir .\ByAttack\ByDSTIP
        :: Get List of attack files
dir .\ByAttack\*.txt /b >.\DirList.tmp
        :: Go process each file in turn
for /f "tokens=1" %%L in ('type .\DirList.tmp') do call :sAFile %%L


pause



        :: Count up the Src Scans
:: dir .\ByAttack\BySRCIP\*.txt /b >.\DirList.tmp
:: for /f "tokens=1" %%M in ('type .\DirList.tmp') do call :sSrcCount %%M
        :: Count up the DST Scans
dir .\ByAttack\ByDSTIP\s*.txt /b >.\DirList.tmp
for /f "tokens=1" %%N in ('type .\DirList.tmp') do call :sDSTCount %%N

        :: Do totals
set /A ETotal=0+0
for /f "tokens=2" %%O in ('type .\ByAttack\BySRCIP\TotalEvents.wri') do call :tot %%O
echo. >> .\ByAttack\BySRCIP\TotalEvents.wri
echo Total Events:      %ETotal% >> .\ByAttack\BySRCIP\TotalEvents.wri
set /A ETotal=0+0
for /f "tokens=2" %%P in ('type .\ByAttack\ByDSTIP\TotalEvents.wri') do call :tot %%P
echo. >> .\ByAttack\ByDSTIP\TotalEvents.wri
echo Total Events:      %ETotal% >> .\ByAttack\ByDSTIP\TotalEvents.wri
if exist *.tmp del .\*.tmp >nul
del .\CurrFile.txt >nul
if exist .\ByAttack\*.tmp del .\ByAttack\*.tmp >nul
now EndRun SortAttackIP.cmd
goto :eof

        :sAFile
                :: %1 - file to process
        set leFileName=%1
        now.exe Processing Attack File  %1
        title Processing Attack File %1
                :: read a line and sub to separate the IP from the port
        for /f "tokens=1-9" %%g in ('type .\ByAttack\%1') do call :sSocket %%g %%h %%i %%j %%k
%%l %%m %%n %%o
        goto :eof

                :sSocket
```

```
                :: reform orginal line
        set leCurLin=%1 %2 %3 %4 %5 %6 %7 %8 %9
        set leIPSRC=%4
        set leIPTRG=%5
                :: Break Source socket into IP and Port, go sub to update files
::      for /f "Delims=: tokens=1,2" %%I in ("%leIPSRC%") do call :sSRCIPLog %%I %%J
                :: Break Destination socket into IP and Port, go sub to update files
        for /f "Delims=: tokens=1,2" %%I in ("%leIPTRG%") do call :sDSTIPLog %%I %%J
        echo   %1 %2 %4 %5 %6 %7 %8 %9
        goto :eof

                :sSRCIPLog
                        :: %1 = Source IP address
                        :: %2 = Source Port

                        :: Kludge - We only need to collect src data from one of the
                        :: PortScan Alert files.
                if /I %leFileName%==tPortscanD.txt goto :eof
                if /I %leFileName%==tPortscanS.txt goto :eof
                        :: Stuff it
                echo %leCurLin% >>.\ByAttack\BySRCIP\s%1.txt
                goto :eof

                :sDSTIPLOG
                        :: %1 = Source IP address
                        :: %2 = Source Port

                        :: Kludge - the PortScan files do not have destination IP's
                        :: skip them
                if /I %leFileName%==tPortscanD.txt goto :eof
                if /I %leFileName%==tPortscanS.txt goto :eof
                if /I %leFileName%==tPortscanE.txt goto :eof

                        :: Break the IP address down to a class C and
                        :: create a directory for each
::              echo %1 > .\env3.tmp
::              for /f "Delims=. tokens=1,2,3" %%K in ('type .\env3.tmp') do set
NetID=%%K.%%L.%%M
::              if not exist .\ByAttack\ByDstIP\s%NetID% mkdir
.\ByAttack\ByDstIP\s%NetID%
                        :: Now update the log
::              echo %leCurLin% >>.\ByAttack\ByDSTIP\s%NetID%\s%1.txt
echo %leCurLin% >>.\ByAttack\ByDSTIP\s%1.txt
                goto :eof

        :sSrcCount
        set FileName=%1
        type .\ByAttack\BySRCIP\%1 |T count > .\LineCount.tmp
        for /f "tokens=1" %%N in ('type .\LineCount.tmp') do call :UpdateSRC %%N
        goto :eof

                :UpdateSRC
                echo %FileName% %1 events counted >>.\ByAttack\BySRCIP\TotalEvents.wri
                echo %FileName% %1 events counted
                goto :eof
```

```
        :sDSTCount
                :: Process each Subdirectory (subnet)
::      set SubNetDir=%1
::      set /A SubCount=0+0
::      set /A NodeCnt=0+0
::      dir .\ByAttack\ByDSTIP\%SubNetDir%\*.txt /b >.\DirList2.tmp
::      for /f "tokens=1" %%i in ('type .\DirList2.tmp') do call :SubNetDSTCount %%i
::      set /a ratio= %SubCount% / %NodeCnt%
::      echo %SubNetDir% %SubCount% events counted for %NodeCnt% nodes (%Ratio%)
>>.\ByAttack\ByDSTIP\TotalEvents.wri
::      echo %SubNetDir% %SubCount% events counted for %NodeCnt% nodes (%Ratio%)

        set FileName=%1
        type .\ByAttack\ByDSTIP\%1 |T count > .\LineCount.tmp
        for /f "tokens=1" %%N in ('type .\LineCount.tmp') do call :UpdateDST %%N
        goto :eof

                :UpdateDST
                echo %FileName% %1 events counted >>.\ByAttack\ByDSTIP\TotalEvents.wri
                echo %FileName% %1 events counted
                goto :eof


                :SubNetDSTCount
                        :: Process each file (node) in each Subdirectory (subnet)
                set /A NodeCnt=%NodeCnt%+1
                type .\ByAttack\ByDSTIP\%SubNetDir%\%1 |T count > .\LineCount.tmp
                for /f "tokens=1" %%j in ('type .\LineCount.tmp') do call :SubTotDST %%j
                goto :eof


                        :SubTotDST
                                :: Keep a running total for each subnet
                        set /A SubCount=%SubCount%+%1
                        goto :eof


        :tot
        set /A ETotal=%ETotal%+%1
        goto :eof


:eof
```

Appendix F - SortOOSIP.cmd

This file sorts the OOS data by source and destination IP

```
:: Program by John Topp
:: ToppJJ@state.gov

@echo off
Title Sorting by OOS Types
if not exist .\ByOOS mkdir .\ByOOS
if not exist .\ByOOS\SRC mkdir .\ByOOS\SRC
if not exist .\ByOOS\TGT mkdir .\ByOOS\TGT
dir .\DataInOOS\*.txt /b >DirList.tmp
:: for /f "tokens=1" %%L in ('type DirList.tmp') do call :sOFile %%L
        :: Count up the Src Scans
:: echo    Counting Src Records
:: dir .\ByOOS\SRC\*.txt /b >.\DirList.tmp
:: for /f "tokens=1" %%M in ('type .\DirList.tmp') do call :sSrcCount %%M
        :: Count up the Tgt Scans
echo    Counting Dst records
dir .\ByOOS\TGT\*.txt /b >.\DirList.tmp
for /f "tokens=1" %%N in ('type .\DirList.tmp') do call :sTgtCount %%N

        :: Do totals
:: set /A ETotal=0+0
:: for /f "tokens=2" %%O in ('type .\ByOOS\SRC\TotalEvents.wri') do call :tot %%O
:: echo. >> .\ByOOS\SRC\TotalEvents.wri
:: echo Total Events:      %ETotal% >> .\ByOOS\SRC\TotalEvents.wri

set /A ETotal=0+0
for /f "tokens=2" %%P in ('type .\ByOOS\TGT\TotalEvents.wri') do call :tot %%P
echo. >> .\ByOOS\TGT\TotalEvents.wri
echo Total Events:      %ETotal% >> .\ByOOS\TGT\TotalEvents.wri

del .\*.tmp /q >nul
now End Run ByOOS.cmd
goto :eof

        :sOfile
        set leCurrFile=%1
        echo now processing OOS File %1
        type .\DataInOOS\%leCurrFile% |T repl '#22' '[22]' > .\tempb.tmp
        type .\tempb.tmp |T repl '#3E' '[3E]' > .\Current.tmp
        set leCapture=T
        for /f "tokens=1,2,3,4,5*" %%i in (.\Current.tmp)  do set leLine="%%i %%j %%k %%l %%m
%%n" &&call :sLine
        goto :eof

                :sLine
                for /f "tokens=1-4" %%p in (%leLine%) do call :sStore %%p %%q %%r %%s
                goto :eof


                        :sStore
```

```
                        if %leCapture%==F goto :bsline
                                :: Key line
::                      for /f "Delims=: tokens=1,2" %%I in ("%3") do set leSrcIP=%%I
                        for /f "Delims=: tokens=1,2" %%J in ("%4") do set leTgtIP=%%J
                        set leCapture=F
                        :bsline
                                :: Save to source
::                      echo %leLine%>> .\ByOOS\SRC\%leSrcIP%.txt
                                :: Save to Destination
                        echo %leLine%>> .\ByOOS\TGT\%leTgtIP%.txt

::                      echo %leTgtIP% > .\env3.tmp
::                      for /f "Delims=. tokens=1,2,3" %%K in ('type .\env3.tmp') do set
NetID=%%K.%%L.%%M
::                      if not exist .\ByOOS\TGT\s%NetID% mkdir .\ByOOS\TGT\s%NetID%
                                :: Now update the log
::                      echo %leLine%>> .\ByOOS\TGT\s%NetID%\%leTgtIP%.txt
                        if %1== + set leCapture=T
                        echo %leCurrFile% SRC=%leSRCIP% TGT=%leTGTIP%
                        goto :eof


        :sSrcCount
        set FileName=%1
        type .\ByOOS\SRC\%1 |T incl '=+' > .\temp1.tmp
        type .\temp1.tmp |T count > .\LineCount.tmp
        for /f "tokens=1" %%t in ('type .\LineCount.tmp') do call :UpdateSRC %%t
        goto :eof

                :UpdateSRC
                echo %FileName% %1 events counted >>.\ByOOS\SRC\TotalEvents.wri
                echo %FileName% %1 events counted
                goto :eof


        :sTGTCount
        set FileName=%1
        type .\ByOOS\TGT\%1 |T incl '=+' > .\temp1.tmp
        type .\temp1.tmp |T count > .\LineCount.tmp
        for /f "tokens=1" %%t in ('type .\LineCount.tmp') do call :UpdateTGT %%t
        goto :eof

                :UpdateTGT
                echo %FileName% %1 events counted >>.\ByOOS\TGT\TotalEvents.wri
                echo %FileName% %1 events counted.
                goto :eof



::      :SUBNET sTgtCount
::              :: Process each Subdirectory (subnet)
::      set SubNetDir=%1
::      set /A SubCount=0+0
::      set /A NodeCnt=0+0
::      dir .\ByOOS\TGT\%SubNetDir%\*.txt /b >.\DirList2.tmp
```

```
::          for /f "tokens=1" %%i in ('type .\DirList2.tmp') do call :SubNetDSTCount %%i
::          set /a ratio= %SubCount% / %NodeCnt%
::          echo %SubNetDir% %SubCount% events counted for %NodeCnt% nodes (%Ratio%)
>>.\ByOOS\TGT\TotalEvents.wri
::          echo %SubNetDir% %SubCount% events counted for %NodeCnt% nodes (%Ratio%)
::          goto :eof

::                  :SubNetDSTCount
::                          :: Process each file (node) in each Subdirectory (subnet)
::                  set /A NodeCnt=%NodeCnt%+1
::                  type .\ByOOS\TGT\%SubNetDir%\%1 |T incl '=+' > .\temp1.tmp
::                  type .\temp1.tmp |T count > .\LineCount.tmp
::                  for /f "tokens=1" %%j in ('type .\LineCount.tmp') do call :SubTotDST %%j
::                  goto :eof

::                          :SubTotDST
                                    :: Keep a running total for each subnet
::                          set /A SubCount=%SubCount%+%1
::                          goto :eof

::                  :SUBNET UpdateTGT
::                  echo %FileName% %1 events counted >>.\ByOOS\TGT\TotalEvents.wri
::                  goto :eof


          :tot
          set /A ETotal=%ETotal%+%1
          goto :eof

:eof
```

Appendix G ByAttack.cmd

This file sorts the alert data by snort alert

```
:: Program by John Topp

@echo off
Title Sorting by Attack Types
if not exist .\ByAttack mkdir .\ByAttack
        :: Get List of attack files
dir .\DataInAttack\*.txt /b >.\DirList.tmp
        :: Go process each file in turn
for /f "tokens=1" %%L in ('type .\DirList.tmp') do call :sAFile %%L
        :: Do a total
dir .\ByAttack\t*.txt /b >.\DirList.tmp
for /f "tokens=1" %%M in ('type .\DirList.tmp') do call :sCount %%M

        :: Clean up
del .\*.tmp /q >nul
del .\CurrFile.txt >nul
goto :eof

        :sAfile
        echo now processing Attack File %1
        copy .\DataInAttack\%1 CurrFile.txt >nul
        call :sProcessAttack WinGate1080.txt "WinGate 1080 Attempt"
        call :sProcessAttack Watchlist220.txt "Watchlist 000220 IL-ISDNNET-990517"
        call :sProcessAttack Watchlist222.txt "Watchlist 000222 NET-NCFC"
        call :sProcessAttack PortscanD.txt "spp_portscan: PORTSCAN DETECTED from"
        call :sProcessAttack PortscanS.txt "spp_portscan: portscan status from"
        call :sProcessAttack PortscanE.txt "spp_portscan: End of portscan from"
        call :sProcessAttack Nullscan.txt "Null scan!"
        call :sProcessAttack NMAP_TCP_Ping.txt "NMAP TCP ping!"
        call :sProcessAttack Broadcast_Ping_to_subnet.txt "Broadcast Ping to subnet 70"
        call :sProcessAttack SMB_Name_Wildcard.txt "SMB Name Wildcard"
        call :sProcessAttack SNMP_public_access.txt "SNMP public access"
        call :sProcessAttack SYN-FIN_scan.txt "SYN-FIN scan!"
        call :sProcessAttack Back_Orifice.txt "Back Orifice"
        call :sProcessAttack External_RPC_call.txt "External RPC call"
        call :sProcessAttack SUNRPC_highport_access.txt "SUNRPC highport access!"
        call :sProcessAttack Tiny_Fragments.txt "Tiny Fragments - Possible Hostile Activity"
        call :sProcessAttack Queso_fingerprint.txt "Queso fingerprint"
        call :sProcessAttack Attempted_Sun_RPC_high_port_access.txt "Attempted Sun RPC high port
access"
        call :sProcessAttack TCP_SMTP_Source_Port_traffic.txt "TCP SMTP Source Port traffic"
        call :sProcessAttack Probable_NMAP_fingerprint_attempt.txt "Probable NMAP fingerprint
attempt"
        call :sProcessAttack Possible_wuftpd_exploit.txt "SITE EXEC - Possible wu-ftpd exploit -
GIAC000623"
        call :sProcessAttack Possible_wuftpd_exploit.txt "site exec - Possible wu-ftpd exploit -
GIAC000623"
        call :sProcessAttack Happy_99_Virus.txt "Happy 99 Virus"
        call :sProcessAttack connect_to_515_from_inside.txt "connect to 515 from inside"
                :: This cleans up known crap in the excess report
        call :sRemoveBS %1
```

```
goto :eof

:: **** Begin Subs

:sProcessAttack

        :: %1 = Filename of text file
        :: %2 = String to search for (will format to %string%)
        :: .\CurrFile.txt = Current file under scrutiny

echo    Looking for event: %2
        :: Convert the string (remove quotes)
echo %2 > .\env1.tmp
type .\env1.tmp |T repl "" " >.\env2.tmp
for /f "tokens=1*" %%J in ('type .\env2.tmp') do set string=%%J %%K
        :: Ensure a capture file does NOT already exist
if exist .\ByAttack\%1 del .\ByAttack\%1
        :: Capture important stuff, strip out excess crap
type .\CurrFile.txt |T incl '%string%' >.\ByAttack\raw1.tmp
type .\ByAttack\raw1.tmp |T repl ' [**] ' ' ' >.\ByAttack\raw2.tmp
type .\ByAttack\raw2.tmp |T repl '-#3E' " >.\ByAttack\raw1.tmp
        :: Replace string - we do this so that we can standardize the
        :: output of the attack file - this way, the first listed
        :: IP is always in the 4th col. This will come in handy later
type .\env2.tmp |T repl ' ' '_' >temp.tmp
for /f "tokens=1*" %%J in ('type .\temp.tmp') do set RString=%%J %%K
type .\ByAttack\raw1.tmp |T repl '%String%' '%RString%' >>.\ByAttack\%1
        :: Clean up
del ByAttack\raw?.tmp /q >nul
        :: Since we have just captured all %string% data out of
        :: the file, let's subtract that same data from the orginal
        :: input file. Eventually, the only data left in CurrFile.txt
        :: will be data that we don't know about. This information will
        :: be preserved in the file EXCESS.TXT when all is said and done.
if exist .\ByAttack\%1 type .\CurrFile.txt |T excl '%string%' >".\ScratchA.tmp
if exist .\ScratchA.tmp del .\CurrFile.txt
if exist .\ScratchA.tmp ren .\ScratchA.tmp CurrFile.txt



        :: Concatenate
call :sConcat %1
goto :eof


:sConcat
        :: %1 = Report to concatenate

set leReport=%1
        :: Check how many lines are in the report
type .\ByAttack\%1 |T Count > .\LinCnt.tmp
        :: Zero lines is bad - go check
for /F "tokens=1" %%Q in ('type .\LinCnt.tmp') do call :sCheck4Zero %%Q %leReport%
        :: Report is already deleted if zero lines
if not exist .\ByAttack\%1 goto :DoneWithConcat
```

```
                :: Check if we have a prior report (t{report name})
        if exist .\ByAttack\t%1 goto :MainIsHere
                        :: No prior, must be first time this "string" has poped up
                        :: Just do a simple rename
                if exist .\ByAttack\%1 ren .\ByAttack\%1 t%1
                goto :DoneWithConcat

                :MainIsHere
                        :: Prior report does exist! Concatenate the new data
                        :: witht the old data
                type .\ByAttack\%1 |T repl '#1A' '' >> .\ByAttack\t%1
                if exist .\ByAttack\%1 del .\ByAttack\%1
        :DoneWithConcat
        goto :eof


:       sRemoveBS
                :: subtract the following lines - we are not interested in them

        type CurrFile.txt |T excl 'gzip' >"ScratchA.tmp"
        if exist "ScratchA.tmp" del CurrFile.txt
        if exist "ScratchA.tmp" ren "ScratchA.tmp" CurrFile.txt

        type CurrFile.txt |T excl '/usr/home' >"ScratchA.tmp"
        if exist "ScratchA.tmp" del CurrFile.txt
        if exist "ScratchA.tmp" ren "ScratchA.tmp" CurrFile.txt

        type CurrFile.txt |T excl '****' >"ScratchA.tmp"
        if exist "ScratchA.tmp" del CurrFile.txt
        if exist "ScratchA.tmp" ren "ScratchA.tmp" CurrFile.txt
        if exist scratchb.tmp del scratchb.tmp

        type CurrFile.txt |T excl 'Snort' >"ScratchA.tmp"
        if exist "ScratchA.tmp" del CurrFile.txt
        if exist "ScratchA.tmp" ren "ScratchA.tmp" CurrFile.txt
        if not exist ByAttack\Excess.wri echo Not Processed >ByAttack\Excess.wri

        type CurrFile.txt |T repl '#1A' '' >> ByAttack\Excess.wri

        goto :eof

        :sCheck4Zero
        if %1 == 0 Del ByAttack\%2
        goto :eof

        :sCount
                :: %1 = Current File to count
        type .\ByAttack\%1 |T count > .\LineCnt.tmp
        for /F "tokens=1" %%R in ('type .\LineCnt.tmp') do set LineCount=%%R
        echo %1 records %LineCount% events >> .\ByAttack\TotalEvents.wri
        goto :eof

:eof
```

Appendix H – T.exe (TEXTools)

TEXTools 1.31 Copyright (c) 1997-1999 Firefly Software All Rights Reserved

TEXTools is a powerful and easy-to-use set of 50+ DOS filters integrated into one EXE that can be combined like building blocks into mini-programs called "pipes" to solve those everyday text-processing problems. TEXTools can be used interactively from the command prompt to quickly handle simple ad hoc requests or from batch files to handle more complex tasks. TEXTools can help you generate reports, interface incompatible software systems, extract data from cumbersome logfiles, automate manual processes, customize text output from other programs, perform text searches, format program listings, convert exported text for use by other software packages, perform base conversions, remove duplicate data, convert fixed-length data to comma-delimited and vice versa, format mailing lists for printing -- all this and more by simply combining filters! Registered users can even create their own user-defined filters from other TEXTools filters! TEXTools includes a built-in debugging tool that enables you to view intermediate text results between filters.

TEXTools runs from either MS-DOS or from Windows 95/98.

This is a FREE version of TEXTools. ALL ELEMENTAL FILTERS ARE AVAILABLE FOR USE. Registration merely entitles you to improved performance and provides you the ability to create your own filters via UDF's, (see documentation).

TEXTools 1.31 Copyright (c) 1997-1999 Firefly Software All Rights Reserved

How To Obtain a Registered Copy of TEXTools

By ordering your own registered copy of TEXTools you will receive the latest version complete with all available filters on a 3.5" diskette along with a 50-page printed manual. As a small token of our appreciation you will also be entitled to use of the UDF option, (see documentation for details).

To download the latest FREE version of TEXTools or to obtain a registered copy of TEXTools at a cost of $35.00 (US), please visit our website at

http://www.FireflySoftware.Com

Direct any e-mail to "Inquiry@FireflySoftware.Com"

Thanks for supporting TEXTools and other shareware products in-general.

Last Updated: October 24th, 2017

Upcoming Training

SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
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 Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
Community SANS Pensacola SEC503	Pensacola, FL	Nov 27, 2017 - Dec 02, 2017	Community SANS
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Munich December 2017	Munich, Germany	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LA	Jan 08, 2018 - Jan 13, 2018	Live Event
Las Vegas 2018 - SEC503: Intrusion Detection In-Depth	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	vLive
SANS Las Vegas 2018	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	Live Event
SANS London February 2018	London, United Kingdom	Feb 05, 2018 - Feb 10, 2018	Live Event
SANS Dallas 2018	Dallas, TX	Feb 19, 2018 - Feb 24, 2018	Live Event
SANS Northern VA Spring - Tysons 2018	McLean, VA	Mar 17, 2018 - Mar 24, 2018	Live Event
SANS Secure Canberra 2018	Canberra, Australia	Mar 19, 2018 - Mar 24, 2018	Live Event
SANS 2018	Orlando, FL	Apr 03, 2018 - Apr 10, 2018	Live Event
SANS Baltimore Spring 2018	Baltimore, MD	Apr 21, 2018 - Apr 28, 2018	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201805,	May 02, 2018 - Jun 07, 2018	vLive
SANS Security West 2018	San Diego, CA	May 11, 2018 - May 18, 2018	Live Event
Community SANS Columbia SEC503	Columbia, MD	Aug 13, 2018 - Aug 18, 2018	Community SANS
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced